UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

INVESTIGAÇÃO E IMPLEMENTAÇÃO DE UMA DATAWAREHOUSE PARA DESCOBERTA FORENSE, CLASSIFICAÇÃO, AGREGAÇÃO E ANÁLISE DE IDENTIDADES E ACESSOS A DADOS SENSÍVEIS

João Miguel Alabaça Ferreira

PROJECTO

MESTRADO EM ENGENHARIA INFORMÁTICA
Especialização em Sistemas de Informação

2015

UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

INVESTIGAÇÃO E IMPLEMENTAÇÃO DE UMA DATAWAREHOUSE PARA DESCOBERTA FORENSE, CLASSIFICAÇÃO, AGREGAÇÃO E ANÁLISE DE IDENTIDADES E ACESSOS A DADOS SENSÍVEIS

João Miguel Alabaça Ferreira

PROJECTO

MESTRADO EM ENGENHARIA INFORMÁTICA
Especialização em Sistemas de Informação

Trabalho orientado pelo Prof. Doutor António Casimiro Ferreira da Costa
e co-orientado por Prof. Doutor Eng. José António dos Santos Alegria

2015

Agradecimentos

Antes de mais quero de agradecer aos meus pais e avós por me terem proporcionado uma excelente educação, assim como, todo o apoio motivacional e financeiro ao longo da licenciatura e mestrado, sendo estes fatores cruciais para que se tornasse possível todo o meu percurso académico.

Gostaria de agradecer aos meus amigos e colegas de curso que trabalharam comigo no desenvolvimento de projetos pela ajuda, confiança, companhia e partilha de conhecimentos e com todos aqueles que direta ou indiretamente me ajudaram a atingir os objetivos durante a minha licenciatura e mestrado em Engenharia Informática. Agradeço em particular ao Hugo Neves, ao Tiago Inocêncio, ao João Oliveira, ao Paulo Ribeiro, ao Roberto Silva, e ao Edgar Montez pela amizade e companheirismo que formámos ao longo do tempo que resultou numa caminhada académica de valor e sucesso com muita dedicação e empenho.

Agradeço ao Professor António Casimiro por todo o apoio prestado, orientação na concretização deste projeto e pela disponibilidade para ajuda e contribuição para o sucesso da tese de mestrado. Agradeço também ao Engenheiro José Alegria por toda a ajuda prestada, pela constante motivação, coorientação e acompanhamento deste projeto, pelos conhecimentos transmitidos e que me proporcionou a atingir, por ter confiado em mim e pela audácia de me propor um ambicioso projeto como este. Ainda, agradeço à equipa do departamento de Segurança e Privacidade da Portugal Telecom, nomeadamente, ao Hugo Silva, Paulo Serrão e Tiago Sequeira, pelo suporte e disponibilidade no desenvolvimento deste projeto.

Aos meus pais e avós

Resumo

Atualmente, as grandes organizações estão dependentes de grandes redes de computadores para o seu funcionamento. Conhecer e controlar os dispositivos que estão ligados nestas redes e respetivas relações e padrões de comunicação existentes é um fator importante para uma correta e segura gestão da segurança de qualquer organização. Este projeto enquadra-se numa área importante da segurança da informação, designada por Identity and Access Management, sendo uma área essencial a qualquer organização especialmente se estiver cotada na Bolsa Norte Americana, devido às exigências da lei "Sarbanes-Oxley Act of 2002" a que estão sujeitas, ou no caso de se tratar de um banco, devido ao cumprimento do "Basel II Framework". Independentemente disso, qualquer grande organização deve garantir a privacidade da informação dos seus clientes e colaboradores, e também salvaguardar a sua informação de atos de fraude que em tempos de crise tem tendência a aumentar. Para uma gestão mais eficaz destas redes é importante analisar, desenhar e implementar uma *datawarehouse* para descobertas forenses, classificação, agregação e análise de identidades e acessos a informação reservada de modo a que seja possível automaticamente detetar, caracterizar, documentar e reportar situações indevidas ou suspeitas. Por exemplo, um acesso de *login* realizado de forma atípica em termos geográficos ou temporais, seria considerado um ato suspeito. O sistema a desenvolver recorre a tecnologias e algoritmos eficientes para lidar com grandes volumes de informação proveniente de diferentes fontes de dados que disponibilizam informação importante referente a contas de aplicações e respetivos acessos realizados pelos utilizadores de uma empresa. Estas tecnologias visam assim melhorar a gestão de identidades e acessos de uma grande organização recorrendo a uma *datawarehouse* que armazena toda a informação, sendo os dados modelados de modo a garantir a escalabilidade do sistema, facilitar a integração de novos dados, tornar as consultas mais eficientes e, acima de tudo, facilitar a tomada de decisão.

Palavras-chave: gestão de identidades, acesso, monitorização, segurança, utilizadores

Abstract

Currently, large organizations are dependent on large computer networks for its operation. Knowing and controlling the devices that are connected to these networks and their respective relationships and communication patterns, is an important factor for proper and safe management of organizational security. This project is part of an important area of information security, called, Identify and Access Management. This area is essential to any organization, especially if it is listed in the North American stock market, due to the demands of the "Sarbanes-Oxley Act of 2002" law, or if it is a bank, due to the fulfillment of the "Basel II Framework ". Nevertheless, any large organization should ensure the privacy of its customers and employees information, and also safeguard the information from acts of fraud that tends to increase in times of crisis. For more efficient management of these networks it is important to analyze, design and implement a data warehouse for forensic discovery, classification, aggregation and analysis of identity and accesses to reserved information so that it is possible to automatically detect, characterize, document and report undue or suspicious situations. For instance, atypical access concerning geography ot time will be considered a suspicious act. The system uses the technologies and develops efficient algorithms to handle large amounts of information from different data sources that provide important data regarding the accesses made by users of the company. These technologies aim to improve the identity and access management in a large organization, using a datawarehouse that stores all information modeling data to ensure system scalability, ease the integration of new data, to allow efficient queries, and, to facilitate the decision making.

Keywords: identity management, access, monitoring, security, users

Conteúdo

Conteúdo **v**

Lista de Figuras **ix**

Lista de Tabelas **xi**

1 Introdução **1**

1.1 Motivação 1
1.2 Objetivos 3
1.3 Metodologia de investigação a utilizar 5
1.4 Enquadramento institucional 6
1.5 Resumo do trabalho realizado 6
1.6 Organização do documento 10

2 Contexto do trabalho **13**

2.1 Enquadramento geral 13
2.1.1 A plataforma Pulso e seus objetivos 14
2.1.2 Arquitetura Pulso 15
2.1.3 Análise do problema 16
2.2 Conceitos fundamentais 21
2.2.1 Segurança informática 21
2.2.2 Gestão de identidades e acessos 22
2.2.3 Resolução de entidades 29
2.3 Análise de números de identificação 40
2.3.1 Número de bilhete de identidade 41
2.3.2 Número de contribuinte 43
2.3.3 Número de colaborador 44
2.4 Ferramentas 44
2.4.1 Ambiente de desenvolvimento 45
2.4.2 Fase algorítmica 45

2.4.3 Armazenamento da informação 46

3 Arquitetura 49

3.1 Descrição do sistema 49
3.2 Modelo de dados 51
3.2.1 Base de dados – PiasaSources 51
3.2.2 Base de dados – InfoTemp 53
3.2.3 Base de dados – PiasaInfo 54
3.3 Arquitetura do sistema 58
3.3.1 Arquitetura de alto nível 58
3.3.2 Arquitetura detalhada 59

4 Concretização 61

4.1 Funções de similaridade 61
4.2 Diagrama de classes UML 63
4.3 Processo de ETL 66
4.3.1 Vantagens de um processo de ETL 66
4.3.2 Análise das fontes de dados 67
4.3.3 Descrição do processo de ETL 70
4.4 Processo de sintetização de identidades 75
4.4.1 Método para deteção de registos duplicados 75
4.4.2 Método para combinação de registos duplicados 83
4.4.3 Exemplos de sintetização de duplicados 89
4.5 Processo de validação de dados 93
4.6 Plataforma PiasaServer 94
4.6.1 Fase I – PreClustering 94
4.6.2 Fase II – Clustering 97
4.6.3 Fase III – PosClustering 97

5 Avaliação 99

5.1 Procedimentos gerais de avaliação utilizados 99
5.2 Ambiente de teste 100
5.3 Configurações de teste 100
5.4 Apresentação de resultados 104
5.5 Discussão sobre os testes realizados 109

6 Discussão e trabalho futuro **113**

6.1 Trabalho futuro 113
6.1.1 Aperfeiçoamento do modelo de dados 113
6.1.2 Expansão dos dados de entrada 114
6.1.3 Melhoramento do processo de sintetização 114
6.1.4 Desenvolvimento de um *front-end* 115

6.2 Dificuldades enfrentadas 115
6.3 Considerações finais 116

A Resultados dos testes apresentados **117**

A.1 *Log* da plataforma PiasaServer 117

Bibliografia **125**

Lista de Figuras

Figura 1.1 - Desenvolvimento Ágil de Software 5
Figura 2.1 - Arquitetura de subsistema de *collectors*, *targets* e *loggers* [1] 15
Figura 2.2 - Cenário de acessos a aplicações 17
Figura 2.3 - Oracle Identity Manager 27
Figura 2.4 - Microsoft Forefront Identity Manager 28
Figura 2.5 - Tivoli Identity Manager 29
Figura 2.6 - Antes e depois da aplicação desta técnica a uma rede 31
Figura 3.1 – Enquadramento do projeto 50
Figura 3.2 - Diagrama EA do sistema de alto nível 51
Figura 3.3 - Base de dados PiasaSources 52
Figura 3.4 - Base de dados InfoTemp 53
Figura 3.5 - Base de dados PisaInfo 55
Figura 3.6 - Arquitetura do sistema de alto nível 59
Figura 3.7 - Arquitetura do sistema detalhada 60
Figura 4.1 - Diagrama de classes UML 65
Figura 4.2 - Processo ETL no contexto do projeto 67
Figura 4.3 - Exemplo de processo ETLL aplicado à pseudo-aplicação AD 70
Figura 4.4 - Tabela de mapeamento lógico 72
Figura 4.5 - Mapeamento lógico de atributos 74
Figura 4.6 – Estado do índice de Id_NM após deteção de duplicados 82
Figura 4.7 - Estado dos índices invertidos (antes de qualquer unificação) 91
Figura 4.8 - Estado dos índices invertidos (após unificação por Id_NM) 92
Figura 4.9 - Estado dos índices invertidos (após unificação por Id_BI) 92
Figura 4.10 - Sistema de ficheiros 95
Figura 5.1 – Identidades sintetizadas (1) 106
Figura 5.2 - Identidades sintetizadas (2) 106
Figura 5.3 - Identidades sintetizadas (3) 106

Figura 5.4 - Identidades sintetizadas (4) 106
Figura 5.5 – Número de identidades por número de unificações..... 107
Figura 5.6 – Número de identidades por número de utilizadores de rede..... 108
Figura 5.7 - Número de identidades por número de endereços de email..... 108
Figura 5.8 - Número de valores por tipo de atributo..... 109

Lista de Tabelas

Tabela 3.1 - Tabela Source_Accounts 54
Tabela 3.2 – Tabela Identity..... 56
Tabela 3.3- Tabela Account 57
Tabela 3.4 – Tabelas List 58
Tabela 4.1 –Atributos FullName..... 73
Tabela 4.2 - Padronização do atributo FullName..... 73
Tabela 4.3 - Lista de registos de identidades 81
Tabela 4.4 – Índice de Id_NM 81
Tabela 4.5 - Índice de Id_BI 81
Tabela 4.6 - Índice de AD_Users..... 81
Tabela 4.7 - Índice de AD_Emails..... 81
Tabela 4.8 - Tabela de códigos de unificação 87
Tabela 4.9 - Ordem de unificação segundo atributos..... 89
Tabela 4.10 – Exemplo 1 de sintetização (antes)..... 89
Tabela 4.11 - Exemplo 1 de sintetização (depois) 90
Tabela 4.12 - Exemplo 2 de sintetização (antes)..... 90
Tabela 4.13 - Exemplo 2 de sintetização (depois) 90
Tabela 4.14 - Exemplo 3 de sintetização (antes)..... 91
Tabela 4.15 - Exemplo 3 de sintetização (depois) 91
Tabela 4.16 - Conjunto de identidades iniciais 91
Tabela 4.17 - Conjunto de identidades sintetizadas 92
Tabela 5.1 - Antes e depois de uma única operação de unificação..... 102
Tabela 5.2 – Antes e depois de várias operações de unificação..... 102
Tabela 5.3 - Medição e evolução da eficiência do algoritmo de unificação 104
Tabela 5.4 – Tempos de execução 105
Tabela 5.5 – Registos processados..... 105
Tabela 5.6 - Identidades processadas 105

Tabela 5.7 – Número de identidades por número de unificações 107
Tabela 5.8 – Número de identidades por número de utilizadores de rede 108
Tabela 5.9 – Número de identidades por número de endereços de *email*..... 108
Tabela 5.10 – Valores processados 109

Capítulo 1

Introdução

1.1 Motivação

Atualmente, as redes de computadores são bastante grandes e complexas, e continuam a aumentar de forma significativa, sobretudo em grandes organizações empresariais. Isto acontece por diversas razões, por exemplo, devido à crescente quantidade de recursos humanos necessários para o funcionamento da empresa ou devido ao aparecimento de novos dispositivos eletrónicos que necessitam de estar continuamente ligados à rede corporativa da organização para operarem. Por estes motivos, torna-se necessário realizar um rigoroso controlo sobre os acessos efetuados pelos utilizadores bem como verificar a veracidade da identidade utilizada, e assim determinar se são legítimos ou não, e caso não o sejam reportar devidamente a situação.

A Lei Sarbanes-Oxley [6], ou também conhecida por SOX, é uma lei americana assinada a 30 de Julho de 2002, pelos Senadores Paul Sarbanes e Michael Oxley. Esta lei está fortemente ligada a empresas que possuem capital na Bolsa de Nova Iorque, e nasceu devido aos vários escândalos financeiros de várias empresas que sofreram enormes prejuízos na altura. Tem como principal objetivo evitar a "fuga" dos investimentos financeiros e o afastamento dos investigadores causada pela insegurança da gestão das empresas, melhorando a gestão a nível financeiro das empresas e ao mesmo tempo garantir a credibilidade na contabilidade, auditoria e segurança da informação confidencial reduzindo assim fraudes, fuga de investidores entre outros. Esta lei permite a criação de mecanismos de auditoria e segurança confiáveis nas organizações de tal forma a mitigar riscos ao negócio, reduzindo ou até mesmo evitando fraudes utilizando meios que as identifiquem quando estas ocorrem.

Desta forma, o departamento de TI (Tecnologia de Informação) tem um papel extremamente importante nas empresas, sendo responsável pela gestão, segurança da informação e sistemas de redes. Este facto é essencial mas não suficiente, sendo também necessário uma boa prática das pessoas em lidar com a informação, e assim, qualquer grande organização deverá respeitar as regras impostas por esta lei. Com vista

a preservar informação estritamente confidencial, manter a segurança da informação (através da criação de acessos ou permissões, partilhas, entre outros) e a garantir a veracidade dos dados, uma empresa necessita de averiguar, implementar e garantir processos de gestão. Ainda, com o aumento dos postos de trabalho numa grande organização empresarial e consequentemente das redes de computadores torna difícil o papel a desempenhar pelos analistas de segurança, pois hoje em dia as empresas possuem uma quantidade exorbitante quer de informação quer de utilizadores e dispositivos eletrónicos ligados à sua rede corporativa, e controlar todos os eventos capturados pode tornar-se um processo obsoleto. Os eventos considerados relevantes numa organização podem se dividir em dois grupos, físicos e lógicos:

- Exemplos de acessos físicos
    - Cartão "X" entrou/saiu no momento "TS" no torniquete "T" no edifício "E";
    - Cartão "X" marcou entrada/saída no momento "TS" no relógio de ponto "P" no edifício "E";
    - Cartão "X" entrou/saiu no momento "TS" na sala "S" no edifício "E".
- Exemplos de acessos lógicos
    - Dispositivo com MAC "M" obteve IP "I" no momento "TS" por DHCP;
    - Dispositivo com MAC "M" e IP "I" no momento "TS" fez *login* de rede com utilizador de rede "UR" (personalidade de rede);
    - Utilizador aplicacional "UA", através de IP "I", falhou *login* na aplicação "A" no momento "TS".

Ainda, do ponto de vista técnico e científico o projeto a desenvolver coloca problemas interessantes, entre outros, como:

a) Definição e caracterização do que se entende por "identidade certa" e quais as suas características identificadoras (em geral, nome oficial, variantes usadas do nome, endereço de *email* oficial, outros emails usados, número de contribuinte, outros números de identificação como BI, passaporte, nº de empregado, entre outros). Definição de mecanismos de agregação das múltiplas personalidades usadas (tipicamente uma por cada *user account* não autentificável numa Active Directory e envolvendo uma variante do nome do utilizador, um número de identificação que pode não ser o número oficial, por exemplo, o número de contribuinte, e até mesmo um endereço de *email* que não é o oficial) por um mesmo utilizador à volta de uma personalidade "mestre" (tipicamente a usada na Active Directory quando existente).

b) Como determinar de forma eficiente e eficaz o grau de semelhança entre variantes de um mesmo nome português?

c) Como determinar que personalidades são afins e a que identidade pertencem?

d) E quando algumas dessas personalidades não são legítimas? Isto é, pertencem a uma outra personalidade mestre?

Desta forma, o projeto a desenvolver consiste na investigação e implementação de uma *datawarehouse* para descoberta forense, classificação, agregação e análise de identidades e acessos a informação reservada de modo a que seja possível automaticamente detetar, caracterizar, documentar e reportar situações indevidas ou suspeitas. Este projeto visa assim melhorar a segurança da informação da organização, nomeadamente, no que diz respeito à deteção de acessos atípicos, quer sejam suspeitos ou ilícitos a sistemas com informação reservada. Enquadra-se numa importante área de Segurança da Informação, designada por IAM (Identity and Access Management) e encontra-se no âmbito da disciplina de PEI (Projeto de Engenharia Informática) do MEI (Mestrado em Engenharia Informática) da FCUL (Faculdade de Ciências da Universidade de Lisboa), e está a ser desenvolvido com supervisão numa grande empresa, denominada Portugal Telecom, empresa relacionada com sistemas de informação e telecomunicações.

1.2 Objetivos

Os objetivos que se pretendem colmatar com a realização deste projeto envolvem a investigação, desenho, implementação e avaliação de um sistema composto por uma *datawarehouse* central. A *datawarehouse* destina-se a armazenar dados referentes a um conjunto de atividades da organização numa base de dados de forma não volátil e consolidada. Após devidamente alocada é possível obter e analisar grandes quantidades de informação, e produzir diferentes relatórios que facilitam futuras tomadas de decisão. Um armazenamento de dados deste tipo vai permitir aos analistas de segurança, onde não exista a obrigatoriedade de autenticação central num serviço de diretório AD (Active Directory) e um utilizador possa, por consequência, possuir diferentes identidades (através de diferentes personalidades operacionais) em diferentes sistemas e aplicações, determinar, em termos forenses, as identidades mais prováveis dos utilizadores responsáveis pelas evidências capturadas de acessos físicos ou lógicos a informação reservada. Além da identificação dos autores dos acessos capturados pretende-se também agregar todas as personalidades operacionais (contas de utilizadores) à volta da sua personalidade mestre (identidade) e sintetizar assim o perfil mais provável de comportamento dessas identidades em termos das contas que usam e dos acessos que efetuam.

Como ponto de partida assume-se que um utilizador antes de aceder a qualquer aplicação ou base de dados terá de se ligar à rede corporativa e assim obter um endereço IP (Internet Protocol) via DHCP (Dynamic Host Configuration Protocol). Este ato produzirá uma primeira evidência de acesso. Depois, ao autenticar-se na AD, se for esse o caso, produzirá uma outra evidência e aqui haverá informação da personalidade mestre usada tal como está registada na AD (nome oficial, *email* oficial, número de identificação oficial, entre outros). Mais tarde se fizer *login* a uma aplicação, mesmo que esta não force a uma autenticação na AD, produzirá mais outra evidência associada a outra personalidade operacional ou *user account* usada. E assim, por diante. Por outro lado, para entrar num edifício da empresa terá de passar por um torniquete de entrada passando o seu cartão de empregado ou de colaborador externo produzindo mais uma evidência associada a outra personalidade mestre (identidade registada com o cartão usado), agora com informação explicitamente georreferenciada (edifício envolvido). E como é possível associar os endereços IP (Internet Protocol, Protocolo de Internet) atrás referidos a edifícios é possível relacionar acessos lógicos com informação de acessos físicos.

O projeto passará pela criação de um modelo de dados onde se possam representar as várias redes de evidências atrás referidas, envolvendo múltiplas personalidades (operacionais e mestre), relações entre endereços IP, relações temporais, relações causais fatuais e relações causais probabilísticas. O projeto deverá depois, sobre este modelo de dados, desenvolver algoritmos para, com base nas redes de evidências, inferir quais as identidades mais prováveis dos utilizadores responsáveis pelas evidências registadas. Se houver uma autenticação na AD, esta inferência será determinística e conclusiva pois a AD neste projeto tem toda a informação necessária e suficiente para identificar o utilizador. Mas se não houver qualquer autenticação na AD então o processo de identificação poderá não ser unívoco importando neste caso sintetizar as características que possam permitir ao analista forense a caracterização do responsável por esses acessos. Tendo sempre em conta que o facto de não haver evidências de autenticação na AD é em si um fato relevante.

Este projeto é uma componente integrante de um sistema maior capaz de fornecer e disponibilizar informação devidamente processada, categorizada, validada e completa num único local central. Outra componente desse sistema desenvolvida em paralelo por outra equipa é responsável pela captura de eventos da rede corporativa, que em conjunto com este projeto será possível identificar o individuo por de trás desses eventos, associando-os às identidades sintetizadas por este projeto. A realização deste projeto pretende assim melhorar o desempenho de um analista de segurança através de um processo sequencial de operações sobre os dados a produzir, proporcionando uma melhor qualidade da mesma e permitindo a identificação das identidades mais prováveis

dos utilizadores responsáveis pelas evidências capturadas de acessos a informação reservada.

Este projeto integrará a plataforma Pulso desenvolvida na Portugal Telecom. No subcapítulo 2.1.3 encontra-se de forma detalhada o problema enfrentado e as soluções sugeridas para alcançar os objetivos propostos.

1.3 Metodologia de investigação a utilizar

A metodologia adotada vai ser Desenvolvimento Ágil de *Software* ou simplesmente Método ágil. Esta metodologia tenta reduzir o risco pelo desenvolvimento de *software* em curtos períodos, designados de iteração, que tipicamente duram menos de uma semana a até quatro. Cada iteração pode ser entendida como um projeto de *software* em miniatura do seu próprio, e inclui todas as tarefas necessárias para introduzir uma nova funcionalidade: planeamento, análise dos requisitos, codificação, teste e documentação. Ao contrário de um processo tradicional, cada iteração não está inteiramente focada em adicionar um conjunto de novas funcionalidades, mas sim desenvolver uma nova versão do produto ao fim de cada iteração, onde será efetuada uma nova avaliação das prioridades do projeto. As fases que constituem esta forma de desenvolvimento de *software* são:

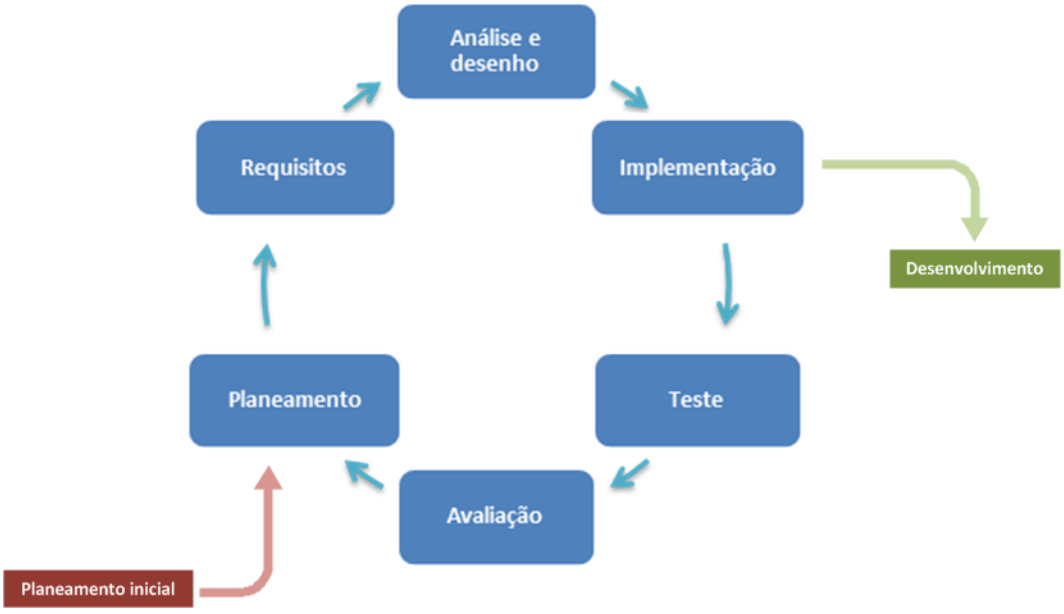

Figura 1.1 - Desenvolvimento Ágil de Software

1.4 Enquadramento institucional

A Portugal Telecom[1] ou também conhecida por Grupo PT foi fundada em 1994 e é uma operadora global de telecomunicações líder a nível nacional em todos os setores em que atua, sendo estes as comunicações fixas, móveis, multimédia e sistemas de informação. A PT Comunicações[2] é uma empresa pertencente ao grupo PT criada em 2000. Esta empresa dispõe de um grande infraestrutura de telecomunicações a nível nacional proporcionando aos seus clientes os mais modernos e recentes meios de comunicação e uma grande diversidade de serviços e soluções que atendem às suas necessidades.

De forma a aumentar os índices de segurança e privacidade do Grupo PT, foi criado o departamento de Segurança e Privacidade da Portugal Telecom, departamento onde se encontra o desenvolvimento deste projeto e onde foi desenvolvido o sistema Pulso.

1.5 Resumo do trabalho realizado

Para alcançar os objetivos propostos foram realizadas várias tarefas indispensáveis para a concretização do projeto. A maioria das tarefas aqui mencionadas foram planeadas e discutidas aquando do início do projeto, mas no entanto, algumas foram refinadas ao longo do desenvolvimento do projeto, e ainda foram adicionadas outras tarefas não planeadas e derivadas de um trabalho de investigação e avaliação, que desta forma, contribuíram para o sucesso do projeto. De seguida, podemos ver a lista de tarefas realizadas durante desenvolvimento do projeto de forma resumida:

- Estudo relativo ao trabalho que tem sido desenvolvido na PT Comunicações que se enquadra no contexto deste projeto:
    - Estudo do trabalho realizado por João Miguel Ribeiro Martins[3] desenvolvido na PT Comunicações, nomeadamente, a leitura da sua tese de mestrado e análise da sua plataforma Pulso/Discovery baseada em Ruby on Rails.
    - Estudo da plataforma Pulso desenvolvida na PT Comunicações, e sua arquitetura para um melhor enquadramento dos seus objetivos e funcionalidades.

[1] http://www.telecom.pt

[2] http://www.ptcom.pt

[3] Investigação e Desenvolvimento de um Sistema Automático de Mapeamento e Visualização de Ativos numa Rede Empresarial de Grande Dimensão

- Análise detalhada do problema a resolver com a concretização deste projeto de forma a compreender melhor os objetivos que se pretendem atingir e como são instanciados atualmente os conceitos de Gestão de Identidades e Acessos nesta organização. E familiarização com as tecnologias a utilizar no projeto, requerendo um estudo inicial sobre a linguagem de programação Ruby e sua variante JRuby.
- Preparação de um ambiente para desenvolvimento e testes deste projeto num servidor dedicado na infraestrutura da organização similar ao que foi desenvolvido pelo João Martins, mas inserido na filosofia de desenvolvimento da equipa da plataforma Pulso, sendo necessário proceder à instalação e configuração de alguns componentes indispensáveis ao projeto. Adicionalmente foram também configurados os repositórios de dados a usar através de uma plataforma interna de gestão de projetos designada por Redmine, o que envolveu também a familiarização com o sistema de controlo de versões designado por Git.

- Estudo de bibliografia fundamental acerca de vários temas para o âmbito da realização do projeto, através da leitura de livros e artigos científicos:
  - Estudo introdutório da área de Segurança Informática com foco na Segurança em Sistemas de Informação com o objetivo de perceber o que se deve monitorizar e quais os métodos utilizados. E estudo da área em que insere o projeto, designada por Gestão de Identidades e Acessos (Identity and Access Management) e a sua terminologia usada. Ainda a análise de soluções existentes no mercado.
  - Investigação e discussão sobre técnicas usadas na área de Resolução de Entidades para aplicação aos registos de *user accounts* (diferentes personalidades operacionais) em diferentes aplicações para os reconciliar num único registo central e canónico (personalidade mestre), e sua relação com o tema de Gestão de Identidades e Acessos. Análise e teste de vários algoritmos usados na importante subárea que é a deteção de registos de duplicados numa base de dados com grandes quantidades de registos, assim como, técnicas de combinação de pares de registos e combinação de campos. Ainda, investigação de métodos para otimizar este processamento reduzindo o tempo de execução.
  - Estudo e análise de várias técnicas e algoritmos de medição das distâncias ou similaridades para nomes portugueses e números, a fim de escolher uma medida eficiente e eficaz para lidar com grandes quantidades de registos.

- Estudo de registos de identidades e contas para as aplicações que vão ser utilizadas no projeto:
    - Análise detalhada e estatística das estruturas e esquemas usados nas bases de dados ou fontes de dados a considerar como *input* (entrada de dados) para determinar que informação contida nestas é útil para o projeto. As bases de dados a usadas e que foram analisadas são: Active Directory com informação acerca das contas da rede corporativa, Pulso/Acessos com informação acerca gestão de utilizadores e acessos de um crescente conjunto de aplicações, plataforma OIM (Oracle Identity Management) usada no provisionamento dos acessos geridos pelo Pulso/Acessos, e das próprias aplicações.
    - Análise dos elementos de informação que caracterizam cada conta, nomeadamente, nomes portugueses, números de Contribuinte, números de Bilhete de Identidade, nomes de empresas, nome de departamentos, entre outros elementos. Estudo de *logs* e *audit trails* relativos à gestão de identidades e contas relevantes para o projeto assim como o formato canónico proposto para a sua representação.
    - Investigação de regras de constituição, importância e formas de validação dos números de identificação usados pela organização, nomeadamente, número de Bilhete de Identidade, número de Contribuinte, Autorização de Residência, Passaporte e número Mecanográfico.
- Criação do modelo dados para aglomerar a rede de evidências integrando as bases de dados auxiliares a usar envolvendo as personalidades operacionais possíveis (*user accounts*) e suas características (variantes de nome, *email*, número de identificação, entre outros) e personalidades mestre (identidades) e suas características (nome oficial, *email* oficial, número de identificação oficial, entre outros) e conjunto de outras personalidades afins:
    - Definição de um formato canónico dos elementos de informação que constitui uma identidade e definição de uma tabela canónica relacional para guardar identidades.
    - Definição de um formato canónico dos elementos de informação que constitui uma *user account* semelhante à tabela de identidades mas adicionalmente com campos para identificação da aplicação, como por exemplo, *user_login* e perfil de origem, e definição de uma tabela canónica relacional para armazenar informação *raw* de todas as *sources accounts* de todas as aplicações.

- Desenvolvimento de algoritmos para processamento dos dados de entrada do projeto:
    - Modelação de forma clara mas de alto nível o funcionamento global do sistema incluindo o modelo de dados definido, e delinear uma arquitetura para suportar todo o projeto. Desenho do modelo de dados através da criação de bases de dados relacionais e respetivas tabelas necessárias para a execução do projeto.
    - Definição das diferentes funções de similaridade aplicadas a diferentes atributos para obtenção de *distance scores*, especificando funções para calcular a similaridade de nomes portugueses, números de identificação e endereços de *email*.
    - Desenvolvimento de rotinas de ETL (Extração, Transformação e Carregamento) para cada fonte de dados, envolvendo a criação de um conjunto regras específicas de transformação ou mapeamento dos diversos atributos. Inicialmente, este processo foi baseado em pequenas extrações em formato CSV (Comma Separated Values) que simulavam as extrações reais. Posteriormente, após a análise das fontes de dados reais procedeu-se à adaptação destas rotinas para trabalhar diretamente com as extrações diárias em formato de tabela relacional.
    - Dada a grande importância de um processo de validação para esta organização, foi elaborado um complexo módulo de funções para a validação de endereços de *email*, números de identificação e datas, de forma a ser utilizado neste projeto e futuramente disponibilizado a outras equipas dentro da organização.
    - Investigação e implementação dos algoritmos necessários para inferir as personalidades mestre (identidades) por detrás dos eventos capturados, e sintetizar o perfil mais provável de comportamento de uma dada identidade em termos das contas que usa e acessos que efetua. Tal como o processo de ETL, também o processo de sintetização de identidades foi aplicado previamente a um ficheiro Excel simples com um conjunto de identidades já sintetizadas mas ainda com imensos duplicados, com o objetivo de testar o algoritmo de inferência a fim de efetuar melhorias de eficácia ao nível da unificação de registos. Posteriormente, este algoritmo foi adaptado para usar dados reais provenientes das rotinas de ETL.
- Realização de testes experimentais da solução desenvolvida e avaliação da escalabilidade, desempenho e eficácia dos resultados produzidos.

- Adaptação da solução desenvolvida para ser executada diariamente a um determinado horário e segundo as normas da equipa Pulso, sendo necessário fazer algumas configurações (nomeadamente familiarização e configuração das gemas Ruby Bundler e Capistrano para realizar o Deploy).

1.6 Organização do documento

Este documento encontra-se composto por seis capítulos e um Apêndice com informação dos resultados dos testes efetuados, e no fim poderemos encontrar a bibliografia utilizada para a realização do mesmo. A estrutura do documento é a seguinte:

Capítulo 2

Descrição do enquadramento da plataforma Pulso desenvolvida pela DSP (Departamento de Segurança e Privacidade) na Portugal Telecom focando a sua arquitetura e serviços que disponibiliza para uma posterior integração deste projeto de forma a responder aos novos requisitos propostos pelos objetivos mencionados em 1.2. Análise de forma detalhada do problema identificando claramente as anomalias que se pretendem colmatar com este projeto, assim como, os aspetos sugeridos para a sua solução. Apresentação de uma visão sobre os estudos mais importantes nas áreas em que se insere o projeto para análise, tratamento e classificação do problema, como são instanciados estes aspetos atualmente na Portugal Telecom, e por último, um estudo detalhado sobre os números de identificação a processar e as tecnologias que suportam a solução e a razão dessas escolhas.

Capítulo 3

Caracterização do sistema, do fluxo de informação e a forma como todos os componentes interagem entre si, baseado nos temas apresentados no capítulo anterior e na análise do problema de forma a alcançar os objetivos do projeto. Definição da arquitetura do sistema.

Capítulo 4

Descrição de todo o trabalho realizado a nível de desenvolvimento aplicacional, documentando detalhadamente os métodos e algoritmos adotados, e as bibliotecas usadas.

Capítulo 5

Apresentação de resultados após a realização de diversos testes ao funcionamento da aplicação em ambiente empresarial, assim como os respetivos dados estatísticos relativamente ao desempenho e aos dados de entrada. Descrição do ambiente e os procedimentos adotados a que a aplicação foi submetida a avaliação em termos funcionais e de desempenho.

Capítulo 6

Conclusão da tese e discussão de alguns aspetos importantes referidos nos capítulos anteriores. Indicação das várias alternativas encontradas para o desenvolvimento de algumas funcionalidades, discussão dos motivos que levaram a escolher as opções tomadas, e as dificuldades encontradas durante a realização do projeto. Ainda, apresentação de um conjunto de possíveis melhorias e otimizações da aplicação desenvolvida num trabalho futuro.

Capítulo 2

Contexto do trabalho

Neste capítulo é apresentado o contexto do trabalho, começando por um enquadramento geral deste projeto na Portugal Telecom descrevendo a atual plataforma Pulso de modo a uma melhor compreensão acerca da sua arquitetura e funcionalidades disponibilizadas. De seguida, é demonstrado como é atualmente feita a Gestão de Identidades e Acessos na Portugal Telecom e o que levou esta organização a investir num projeto *open source* nessa área, e logo depois é discutido detalhadamente o problema a tratar bem como a solução proposta para a realização deste projeto.

Após a especificação dos objetivos a atingir foi fundamental realizar uma investigação e leitura prévia para a familiarização da terminologia, conceitos e procedimentos utilizados nas áreas envolventes deste projeto, sendo elas, Segurança Informática, Gestão de Identidades e Resolução de Entidades. Ainda, foi realizado um estudo para conhecer melhor a estrutura e métodos de validação de um conjunto de documentos de identificação usados na organização como elementos identificativos das várias contas de utilizadores. Por último são apresentadas as tecnologias escolhidas para o desenvolvimento deste projeto e os seus motivos.

2.1 Enquadramento geral

A PT Comunicações tem vindo ao longo do tempo a interessar-se pela monitorização e análise da qualidade de risco das suas infraestruturas, sistemas e processos de suporte aos Sistemas de Informação. Desta forma, a Portugal Telecom desenvolveu um programa de projetos designado por Pulso que implementa uma plataforma aberta de desenvolvimento incremental de subsistemas capazes de satisfazer a longo prazo as necessidades acima referidas. O Pulso é um projeto 100% *open source* na área de sistemas de monitorização tendo por objetivo avaliar o desempenho de tecnologias *open source*, e para isso convencionou-se que, tanto a plataforma, como os módulos a desenvolver fossem do tipo LAMP/R (Linux, Apache, MySQL, PHP/Perl e Ruby) e utilizassem outros packages *open source* de grande popularidade, nas áreas de

monitorização e supervisão de sistemas e redes, como por exemplo, Nagios, Snort e TCPDump [1].

Ainda neste capítulo é apresentado de forma detalhada um novo módulo que resultará do desenvolvimento deste projeto e será inserido nesta plataforma Pulso com o objetivo de trazer novas funcionalidades sobre uma área considerada crítica e que não é atualmente monitorizada por esta.

2.1.1 A plataforma Pulso e seus objetivos

O termo pulso refere-se ao programa de projetos, à plataforma técnica que o suporta e ao portal que permite a comunicação com todos os utilizadores. Um dos objetivos para os quais a plataforma Pulso foi desenvolvida inicialmente foi para detetar, armazenar, analisar e comunicar eventos técnicos considerados relevantes, ao nível da qualidade técnica, risco técnico e segurança dos principais SI/TI's (sistemas e infraestruturas informáticas) de suporte ao funcionamento da empresa. Desta forma, a plataforma proporcionou um melhor suporte de incidentes informáticos assim como uma rápida deteção e reação a estes quando mecanismos de prevenção tenham sido insuficientes. Outro objetivo consistiu em implementar e monitorizar processos críticos ao controlo dos sistemas de informação e processos de suporte à função SI/TI da PT Comunicações, como por exemplo, gestão do *backlog* de desenvolvimento, pedidos de acesso a sistemas e infraestruturas, entre outros, controlando assim a qualidade e eficiência destes processos.

A plataforma Pulso, ao nível da prevenção à ocorrência de incidentes permite detetar e registar de forma automática e contínua, todos os utilizadores das infraestruturas e sistemas da Portugal Telecom, e ao mesmo tempo obter informação acerca dos seus níveis de autorização de acessos, local de onde foi realizado o acesso, dispositivos que foram utilizados no acesso (por exemplo, *desktops*, *notebooks* e *tablet* PC's), que informação foi acedida e que perfil foi usado, e também o respetivo *score* de risco. Ainda ao mesmo nível, permite obter e manter de forma automática e contínua o *scoring* de risco técnico de todos os sistemas de informação críticos e respetivas infraestruturas tecnológicas, relativamente à sua QoS (qualidade de serviço), ao seu nível de risco em termos de segurança e também em termos de fiabilidade.

Por outro lado, ao nível da deteção e reação rápida à ocorrência de incidentes, a plataforma Pulso permite detetar em tempo útil incidentes relevantes ao nível de falhas técnicas, ataques externos ou internos, acessos ilícitos a sistemas com informação reservada e uso indevido de recursos. Finalmente, ao nível da interface, a plataforma Pulso suporta os utilizadores dos sistemas de informação da Portugal Telecom através do seu portal. O portal Pulso consiste num meio formal, organizado e centralizado de:

- Comunicar com toda a organização, onde se poderão encontrar todos os conteúdos relevantes para a função de Gestão de Risco Técnico, Segurança e Controlo dos Sistemas de Informação, nomeadamente, normas e políticas de segurança, acessos a sistemas, qualidade técnica e indicadores operacionais diversos.
- Acesso a aplicações de controlo da função SI, como por exemplo, gestão do *backlog*, gestão de pedidos de suporte, gestão de pedidos de acesso, entre outros.

2.1.2 Arquitetura Pulso

O primeiro módulo desenvolvido para a plataforma Pulso foi um subsistema distribuído de *collectors* e *targets* (figura 2.1) destinado a realizar medições de eventos com o objetivo de medir parâmetros de qualidade de rede ponto a ponto, entre locais considerados críticos para a Portugal Telecom. Este subsistema permite assim a prevenção de incidentes e também a sua deteção, pois consiste numa fonte de eventos que são produzidos que servem para detetar falhas de conetividade e poderão ser utilizadas para análise e controlo da QoS da rede.

Devido à grande distribuição geográfica dos locais de acesso (em geral, pontos de venda, *callcenters* e lojas) e dos sistemas da Portugal Telecom foram definidos inicialmente dois tipos de componentes: *collectors* (sondas de rede) e *targets* (agentes de sistema), e posteriormente, um terceiro designado por *loggers*. Adicionalmente, o sistema gera e envia alarmes às entidades que aparentam estar num estado de alerta.

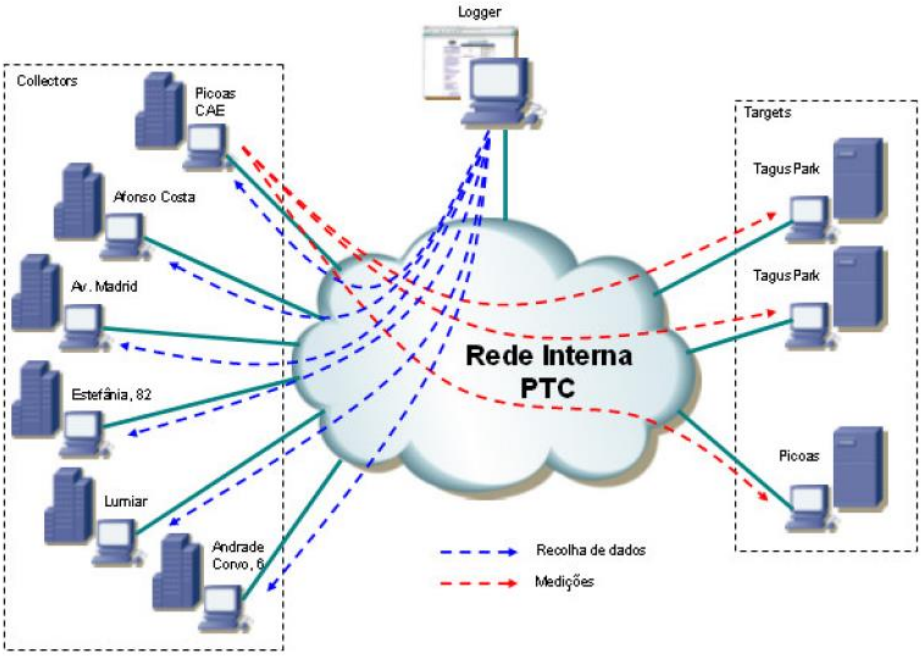

Figura 2.1 - Arquitetura de subsistema de *collectors*, *targets* e *loggers* [1]

Os componentes do tipo *target* são colocados perto dos sistemas cuja QoS se pretende monitorizar e medir. Por outro lado, os componentes do tipo *collector* são colocados em locais que sejam interessantes do ponto de vista do negócio, ao nível de número de utilizadores e das aplicações que usam. Assim, cada *collector* realiza medições relativamente a um ou mais *targets*, medindo parâmetros de rede no troço compreendido entre ele e cada target. De tal forma, cada target apenas responde aos pedidos de medição dos *collectors*.

Os *loggers* visam recolher os resultados das medições efetuadas pelos *collectors* que se encontram armazenados em máquinas definidas para o efeito, designadas por *datasources* e que são responsáveis por guardar os dados na base de dados Pulso. Cada *collector* destina-se a realizar medições para os targets para os quais foi programado e armazena os eventos de forma local no formato canónico do Pulso. Ainda, cada *collector* encontra-se registado como *datasource* no Pulso e por isso o módulo de recolha armazena os eventos medidos no repositório de eventos do Pulso.

2.1.3 Análise do problema

Numa qualquer organização, uma gestão ineficaz de utilizadores e dos seus privilégios de acesso permite que os sistemas fiquem expostos e vulneráveis a ataques externos ou até mesmo internos. Por isso, é necessário proteger o acesso principalmente aos locais onde a informação é considerada altamente reservada utilizando técnicas eficazes. Na Portugal Telecom deve-se proceder a uma gestão de identidades e contas (e perfis de acesso) a este tipo de informação, sobretudo, quando não existe a obrigatoriedade de autenticação central numa AD (Active Directory) e um utilizador possa, por consequência, possuir diferentes identidades em diferentes sistemas e aplicações.

Uma Active Directory consiste num serviço de diretório que guarda informações sobre objetos numa rede de computadores, e disponibiliza essas informações a utilizadores e administradores dessa rede. A AD surgiu da necessidade de existir um único diretório, ou seja, em vez de um utilizador ter uma senha para aceder o portal da empresa, uma senha para ler *emails* do Exchange, uma senha para aceder a um computador, e várias outras senhas, apenas necessita de ter uma única senha para aceder a todos os recursos disponíveis na rede. A AD pode ser definida como uma base de dados que armazena todas as informações que permitem controlar o acesso dos utilizadores à rede, como por exemplo, nomes e senhas dos utilizadores, respetivas permissões de acesso a ficheiros, computadores e impressoras, entre outras.

Na figura 2.2, podemos visualizar um exemplo de um cenário que pode acontecer devido à ausência de uma plataforma de Gestão de Identidades e Contas, onde dois

diferentes utilizadores tentam aceder a duas aplicações distintas utilizando métodos diferentes.

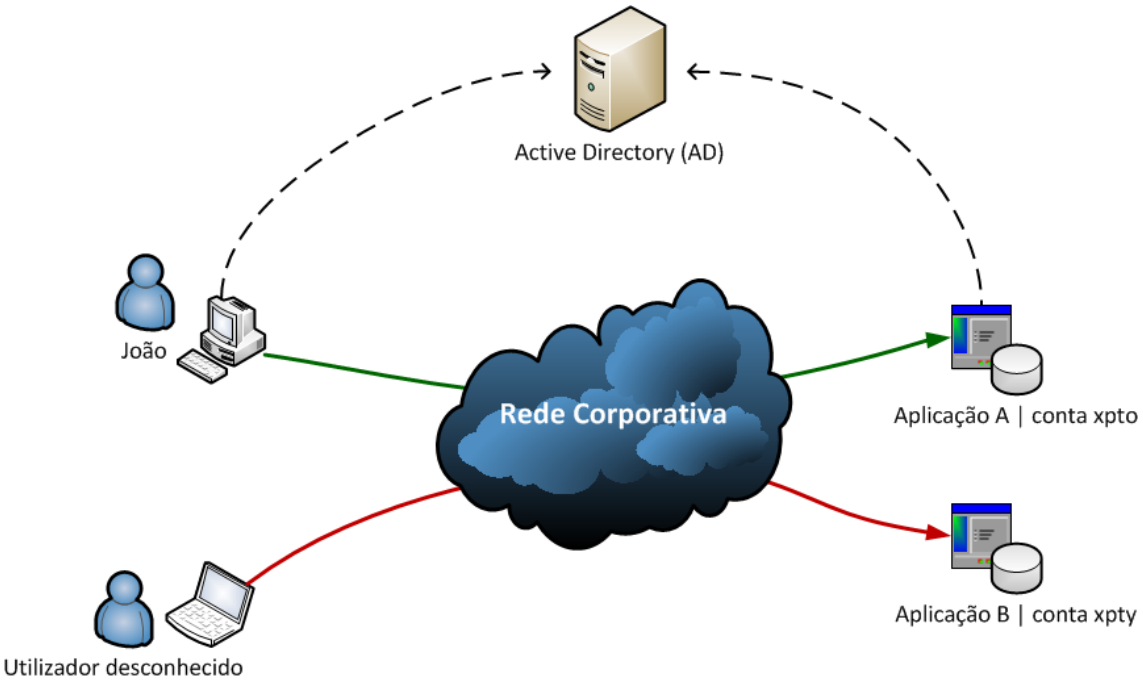

Figura 2.2 - Cenário de acessos a aplicações

Neste cenário, o utilizador João para aceder à aplicação "A" utiliza um computador da organização, e desta forma, são requeridas as suas credências de acesso à rede (provenientes da AD) para aceder ao computador, e de seguida, consegue entrar na rede corporativa. Após isto, e uma vez que a aplicação força a autenticação na AD (isto é, as credenciais da conta "xpto" são as mesmas do acesso à rede), o utilizador como anteriormente se autenticou na AD entra diretamente na aplicação. Por outro lado, o utilizador desconhecido ao contrário do utilizador João utiliza um computador pessoal (em geral, um *notebook*), no qual não é necessário fornecer quaisquer credências para aceder à rede corporativa. De seguida, para aceder à aplicação "B" apenas necessita de conhecer as credenciais da conta "xpty".

Analisando este cenário, podemos concluir que qualquer utilizador que não se autentique na AD, quer para aceder à rede interna quer para aceder a um determinado serviço (por exemplo, uma aplicação), pode passar-se por uma outra pessoa que não o proprietário do acesso, utilizando as credenciais de acesso de um utilizador a um serviço, e assim poderá aceder a informação estritamente confidencial. No caso do utilizador João, como se autenticou na AD, quaisquer outros acessos que realize de forma ilícita, ou seja, utilizando credenciais de outro utilizador, serão detetadas e o autor desses acessos será identificado, pois a AD tem toda a informação necessária e suficiente para identificar o utilizador. Em relação ao utilizador desconhecido, como

nenhuma evidência se refere a uma autenticação na AD, torna mais complicado o procedimento para identificar o autor dos respetivos acessos.

As credenciais de acesso a um serviço são pessoais e devem ser intransmissíveis. É considerado legal uma identidade possuir várias contas mas não uma conta ser partilhada por um grupo de diferentes utilizadores. Por exemplo, um utilizador “A” não pode “emprestar” suas credenciais a outros utilizadores. Da mesma forma, um utilizador "B" não pode usar as credenciais que não lhe pertencem. Ambos os casos são considerados atos ilegítimos que se pretendem detetar.

Na plataforma Pulso, a aplicação de gestão de utilizadores e pedidos de acessos a um conjunto de aplicações é designada por Pulso/Acessos. Esta aplicação é considerada uma plataforma *web* usada para criar novas contas, uma solução de *ticketing*, e ainda um *front-end* entre os requisitantes e os sistemas de *account provisioning* (aprovisionamento de contas). Os sistemas de aprovisionamento de contas permitem a criação de uma conta para uma determinada aplicação e identidade de rede e gestão de direitos da conta para permitir e controlar o seu acesso aos recursos geridos pela aplicação. No módulo Pulso/Acessos encontramos pedidos de abertura de contas aplicacionais, de bases de dados e de sistema, e no seu *backlog* (histórico) fica registada toda a informação acerca de cada pedido, como por exemplo, a identificação do requisitante da conta, a identificação do destinatário da conta e o nome do sistema a aceder.

A longo prazo, a gestão de contas e perfis de utilizadores tende geralmente a tornar-se mais extensa e difícil quando o número de utilizadores e dispositivos aumenta numa rede. Esta situação piora ainda quando a arquitetura de rede necessita de obedecer a determinadas normas de conformidade e controlos de auditoria provenientes de políticas de segurança, como por exemplo a lei Sarbanes-Oxley. Atualmente, na Portugal Telecom o processo de gestão e deteção de acessos recorre a um processo totalmente manual desempenhado por recursos humanos. Esta operação consiste na análise de diversos ficheiros em formato Excel com informação referente a contas e respetivos acessos, e é realizado a cada três meses sempre feito de raiz, ou seja, nunca excluindo o trabalho feito anteriormente. Desta forma, podemos ver que esta solução não é garantidamente precisa e torna-se obsoleta quando a velocidade de aparecimento de nova informação é elevada.

A plataforma de gestão de identidades e acessos usada na Portugal Telecom, OIM (Oracle Identity Management), guarda informação sobre identidades incluindo toda a informação relevante sobre um utilizador, já que o OIM é a plataforma que o Pulso/Acessos usa quando se pretende criar novos acessos (contas) para determinadas aplicações, mas há ainda bastantes aplicações relevantes cujas aberturas de contas não

passam pelas plataformas Pulso/Acessos, e consequentemente, pela plataforma de Gestão de Identidades e Acessos (OIM). A regra geral consiste que em cada conta criada através do Pulso/Acessos deverá existir um registo referente a essa mesma conta no OIM. Se tal não acontecer, significa que o procedimento usado na criação não seguiu as normas formais. Esta situação pode acontecer por que algumas contas poderão ser criadas diretamente através de consolas ou devido a algumas aplicações não requisitarem a utilização do Pulso/Acessos para a criação de novas contas, não sendo aprovisionadas pelo método legal, e assim a plataforma OIM não reúne informação suficiente acerca de todos os utilizadores que possuem contas. Desta forma, a plataforma Pulso não dispõe de qualquer mecanismo que permita à equipa de segurança determinar, em termos forenses, as identidades mais prováveis dos utilizadores responsáveis pelas evidências capturadas de acessos a informação, e detetar em tempo real casos de acessos considerados ilícitos. No entanto, é totalmente legítimo existir registos na plataforma OIM que não possuam contas criadas.

Com estes diferentes métodos de criação de acessos (contas) torna-se necessário implementar um processo de resolução de registos de *user accounts* (diferentes personalidades operacionais) em diferentes aplicações para os reconciliar num único registo central e canónico, designados por personalidades mestre. Este processo incide sobretudo na operação de unificação das diversas contas de várias aplicações e respetivas características que referem uma mesma identidade.

Descrição da solução

Com base nos aspetos referidos acima, a Portugal Telecom decidiu desenvolver um sistema de gestão de identidades e acessos utilizando exclusivamente tecnologias *open source* em vez de adquirir um sistema de IAM (Identity and Access Management) porque devido à dimensão desta empresa, esta implementação levaria a um elevado investimento e tempo a incorporar esse sistema em toda a infraestrutura.

Desta forma, o projeto que se pretende desenvolver vai melhorar e automatizar o procedimento atual para serem detetadas automaticamente situações anómalas ou suspeitas relativamente a acessos a informação privilegiada. O projeto irá colmatar as funcionalidades acima referidas ausentes na organização através da criação de uma *datawarehouse*, na qual representará as várias redes de evidências, quer sejam físicas ou lógicas, envolvendo múltiplas personalidades operacionais (contas de utilizador) e personalidades mestre (identidades), e respetivos relacionamentos.

A *datawarehouse* permitirá também, entre outros:

- Produzir relatórios de dados, analisar grandes volumes de dados e obter informações estratégicas que facilitam a tomada de decisão da organização;

- Aperfeiçoar a criação, manutenção e gestão das identidades com o objetivo de melhorar a qualidade de informação associada a estas, e dos acessos por parte dos utilizadores a aplicações;
- Enriquecer e completar as identidades da plataforma OIM da organização, agrupando num único local centralizado todas as identidades existentes e respetivas contas de utilizador;
- Proceder a uma correta e completa identificação do proprietário das contas existentes quer sigam ou não as normas formais e legais de criação;
- Identificar utilizadores que partilhem os mesmos elementos de informação, por exemplo, o mesmo número de Bilhete de Identidade, número de Contribuinte, entre outros;
- Melhorar a administração dos sistemas de rede das infraestruturas, reduzindo e controlando quaisquer atos de acesso indevido de leitura de informação ou até mesmo o roubo de informação;
- Reduzir o custo de exploração nestas áreas de segurança;
- Salvaguardar a informação (principalmente a mais crítica) de atos criminosos que em tempos de crise poderá aumentar.

Adicionalmente, e durante o desenvolvimento do projeto verificou-se que para além dos objetivos inicialmente previstos também era possível e vantajoso alcançar um outro objetivo. Este objetivo secundário consistiu em ajudar a PTSI (Sistemas de Informação da Portugal Telecom) a melhorar a qualidade dos dados de origem (nomeadamente os que caracterizam as contas de utilizador) utilizando adequados procedimentos de limpeza e tratamento dessa informação, com base num módulo de validação de dados desenvolvido neste projeto. Isto permite assim melhorar a qualidade da informação trocada nesta organização através deste módulo, atribuindo códigos de erros à informação que se pretende validar para posteriormente serem corrigidos e assim reduzir anomalias e "ruído" encontrado nos dados, que eventualmente foram colocados ou não de forma errada. Esta operação não pretende corrigir esta informação mas sim apenas indicar o possível erro resultando da validação para posteriormente outra equipa proceder à sua correção, e desta forma todos os valores origem não são alterados mas sim preservados.

O principal objetivo e foco deste projeto é o agrupamento consolidado das várias contas de utilizadores disponíveis para sintetizar as identidades num único local central e respetivas contas de acesso, sem a existência de duplicados. Os principais *stackholders* ou consumidores dos resultados produzidos por este projeto são:

- Apoiar a equipa responsável pela gestão e captura de eventos para descobrirem as identidades verdadeiras por de trás de um determinado acesso ou ato de *login*, associando um evento a uma conta através dos atributos "account_id" ou "account_login", sendo assim "pendurados" estes eventos às identidades sintetizadas por este projeto. Da sequência desta associação será possível detetar contas promíscuas, identificando a identidade do proprietário da conta e o conjunto de utilizadores que a utilizam indevidamente.
- Apoiar as equipas da PTSI a corrigir informação diretamente nas fontes de dados removendo eventuais discrepâncias e incoerências existentes, e assim elevar a qualidade de informação.

2.2 Conceitos fundamentais

Uma vez que este projeto tem como objetivo inovar a plataforma Pulso existente, adicionando um novo módulo com novas funcionalidades que melhoram a forma de navegação pela informação através de visualizações úteis que permitem representar os atributos de rede relevantes através dos seus componentes e suas relações torna-se necessário fazer um estudo sobre os conceitos e técnicas fundamentais que terão um papel importante no desenvolvimento da proposta apresentada. Desta forma, são apresentados os estudos mais importantes e considerados elementos chave na implementação de qualquer sistema seguro nas áreas de Segurança Informática, Gestão de Identidades e Acessos, e ainda Resolução de Entidades.

2.2.1 Segurança informática

A Segurança Informática ou Segurança de Computadores refere-se à segurança de sistemas onde computadores e redes informáticas são componentes importantes ou estão envolvidos de alguma forma. Hoje em dia, as organizações dependem cada vez mais de infraestruturas informáticas, levando alguns sistemas de informação e algumas redes de computadores a serem bastante críticas, podendo originar, por exemplo, a perda de receitas ou a perda de vidas humanas.

Assim, a segurança nas organizações deve ser uma preocupação na fase de especificação de qualquer sistema de informação ou de qualquer rede. Desta forma, os administradores devem desde o início compreender que é preciso investir em segurança, pois muitas vezes o problema é ignorado por representar um elevado custo inicial, apesar que no fim custa muito mais ainda. Alguns problemas dos sistemas informáticos estão relacionados com os respetivos ciclos de vida muito curtos e a imaterialidade do *software* criando a impressão que é simples. Para colmatar este problema, as organizações devem treinar os seus utilizadores em tecnologias de segurança e tornar

um hábito fazer análises de risco dos sistemas informáticos [4]. A segurança informática deverá ser levada a sério visto que a nossa sociedade está cada vez mais dependente da Informática, nomeadamente, dos sistemas de informação e das redes de computadores.

Dentro desta área de segurança existe uma grande subárea que está relacionada com a segurança da informação ou dados, conhecida por Segurança da Informação. Esta área consiste na operação de proteger um conjunto de informação com o objetivo de salvaguardar o seu conteúdo e preservar o seu valor para um indivíduo ou organização, minimizar eventuais danos e maximizar assim a receita da atividade do negócio [23]. Segundo a ISO[4] (Organização Internacional de Normalização) devem ser considerados quatro princípios básicos:

- Confidencialidade – proteção contra acesso ou leitura a informação;
- Integridade – proteção contra alteração de informação;
- Disponibilidade - proteção contra recusa de provisão ou acesso provocada por intrusos (por exemplo, utilização exagerada de recursos de rede);
- Autenticidade – proteção contra personificação de intrusos.

Na Segurança de TI's (Tecnologias de Informação), a área que lida com identidades e contas de acessos quer seja informação reservada ou não, geralmente armazenada em bases de dados e ficheiros, é designada por Gestão de Identidades e Acessos.

2.2.2 Gestão de identidades e acessos

A Gestão de Identidades ou IdM (Identity Management), ou também conhecida por, Gestão de Acessos e Identidades ou IAM (Identity and Access Management) trata-se de uma área ligada ao processo de identificação, autenticação, autorização, e privilégios ou permissões de acesso nas redes de computadores, tendo como objetivo aumentar a segurança e produtividade enquanto o custo, o tempo de inatividade e duração das tarefas diminuem. Esta área surgiu da necessidade não só de gerir autorizações de acessos mas também de gerir os próprios acessos e o ciclo de vida destes, dentro das organizações [13, 14, 15].

Nesta área, estão abrangidas questões, tais como qual a forma como os utilizadores estão associados a uma identidade, que mecanismos são utilizados na proteção dessas identidades e quais as tecnologias que suportam essa proteção (em geral, protocolos de

[4] Organização Internacional de Normalização consiste numa entidade composta por representantes de várias organizações nacionais que reune normas e padrões de vários países

rede, certificados digitais, *passwords*, entre outros), e são utilizadas tecnologias e serviços, como por exemplo, Active Directory, Service Providers, Web Services, Access Control, Digital Identities, Password Managers, Single Sign-on, Security Tokens, OAuth, entre outros. Atualmente, as grandes e pequenas organizações podem encontrar no mercado soluções ao nível de gestão de identidades e acessos, que relacionam a gestão de acesso e das *passwords* com as identidades, autenticação e autorização, como sistemas, produtos, aplicações e plataformas de gestão de identidades.

Antes de passarmos à descrição destas soluções de gestão de identidades e acessos vamos primeiro compreender os requisitos e a terminologia fundamental usada em sistemas deste tipo.

Identidade

O conceito de identidade refere-se a um conjunto de características essenciais que distinguem uma pessoa ou comunidade. A identidade pode ser construída através da combinação de vários atributos que podem ser hereditários ou inatos, atribuídos, temporários ou não. Do ponto de vista do sistema, uma identidade é um conjunto de atributos que identifica uma entidade de rede. Desta forma, uma identidade pode ser definida como um conjunto de informação sempre atualizado, organizado e acessível através de meios apropriados, e a sua informação pode ser pública, ou seja, ser conhecida por elementos exteriores, como por exemplo, pessoas ou aplicações.

Autenticação

A autenticação consiste na confirmação de alguém ou algo como genuíno, reivindicando a autoria ou a veracidade de alguma coisa. Esta operação também se encontra relacionada com a verificação de uma identidade por uma entidade com base nas suas credenciais de acesso. Na área de Segurança da Informação, a autenticação é um processo que verifica a identidade digital (informação de identificação pessoal exposta numa rede) de um utilizador de um sistema, normalmente, aquando da requisição de um evento de *login* ou acesso num computador ou programa de *software*. O controlo de acessos, normalmente a aplicações, sistemas e bases de dados, trata da confirmação da identidade de uma pessoa ou programa, identificando a sua origem e garantindo que a identidade é quem diz ser.

Desta forma, a autenticação é um componente fundamental em qualquer sistema de controlo de acessos, tendo em conta que permite o acesso a algo para o qual um utilizador foi autorizado. Ao contrário da identidade de um utilizador que deverá ser conhecida por todos, a autenticação de um utilizador deverá ser apenas do conhecimento da identidade, ou seja, privada.

Autorização

Ao contrário da autenticação, que consiste no processo que verifica a identidade de uma pessoa, o processo de autorização verifica se essa pessoa possui privilégios adequados para executar determinadas ações. Desta forma, a autorização realiza-se sempre após a autenticação, e é também uma componente fundamental no controlo de acessos, juntamente com a autenticação. A autorização permite que uma identidade obtenha o direito de acesso a um determinado recurso. Assim, deverão estar definidas previamente políticas de acesso e regras de controlo de acessos, para que seja possível determinar se os pedidos de acesso devem ser aceites ou negados, e caso afirmativo que tipos de privilégios deverão ser aplicados.

Gestão de acessos

A gestão de acessos refere-se à gestão do ciclo de vida de um acesso, ou seja, ao conjunto de processos que permitem controlar as identidades digitais numa rede. O ciclo de vida de um acesso é constituído pelos seguintes processos:

- Provisionamento ou atribuição da identidade;
- Sincronização das identidades na infraestrutura;
- Gestão dos atributos, das credenciais de acesso e dos direitos da identidade;
- Eliminação do provisionamento ou remoção do acesso.

A gestão de acessos permite assim monitorizar os acessos acompanhando todo o ciclo de vida de um determinado acesso, as respetivas regras de acesso, o acesso das identidades e o cumprimento com as normas de segurança definidas.

Sistemas de gestão de identidades e acessos

Como vimos até aqui, a Gestão de Identidades engloba todo o processo de identificação, autenticação e autorização nas redes informáticas. Entende-se como um conjunto de mecanismos para o tratamento e manipulação de identidades, que lida com questões do tipo, como são atribuídas as identidades aos utilizadores, como estão essas identidades e quais as tecnologias de suporte a essa proteção.

Assim, um sistema deste tipo tem como objetivo determinar e gerir os perfis e privilégios dos utilizadores, e controlar os acessos dos utilizadores à informação considerada reservada numa organização. Desta forma, a implementação de um sistema de Gestão de Identidades e Acessos deve satisfazer os seguintes requisitos [7]:

a) Determinar a identidade
    - Relacionar um número identificativo ou um nome a um individuo;

- Voltar a determinar a identidade no caso de existir nova informação associada ao individuo.

b) Descrever a identidade

- Aplicar um conjunto de atributos informativos a um objeto de identidade;
- Atualizar a identidade, alterando um ou mais desses atributos.

c) Seguir a atividade da identidade

- Registar todo o ciclo de atividade de atividade, indicando também as eventuais relações com outras identidades;
- Sintetizar o comportamento das identidades através de padrões.

d) Remover a identidade

- Eliminar a identidade e os respetivos atributos informativos.

Implementação de uma solução de gestão de identidades e acessos

Quando uma organização pretende implementar um produto de Gestão de Identidades existem várias preocupações que deverão ser analisadas com maior rigor. A garantia do seu sucesso não depende apenas que o produto permita integrações com outros serviços usados na organização, ou seja, não depende apenas da instalação e relação do produto na infraestrutura local, implementando mecanismos adicionais, por exemplo, baseado em diretórios de serviços ou em sistemas de Single Sign-On (sistemas que permitem aceder a várias serviços autenticando apenas uma única vez, não sendo necessário autenticar a cada entrada de um serviço) sobre aplicações ou ambientes existentes na organização.

Antes de qualquer empresa adquirir um produto de IdM, é fundamental que faça previamente uma análise da arquitetura e infraestrutura de rede atual da organização de forma a ser capaz de desenvolver e adaptar o produto à situação atual da empresa. Para isto, é necessário preparar os programadores das áreas de TI da organização para tornar o produto de IdM abrangente a todas as aplicações, ou caso contrário, o produto sofrerá um problema de adaptação. Outra grande preocupação está relacionada com a não consideração do seu legado histórico de situações mais antigas e atuais que existem nas organizações, e implementar o produto apenas para as situações futuras, resultando em eventuais inconsistências e erros na futura infraestrutura ou sistemas [7].

Assim, para uma boa implementação de um produto de IdM deve-se previamente realizar uma análise da infraestrutura e tornar o produto abrangente a todos os recursos para adaptar o produto à realidade da organização, resultando num produto sem falhas. Outra preocupação é a complexidade do produto e a exigência de desenvolvimento na

sua implementação, requerendo estar envolvidas todas as áreas da empresa para serem consideradas todas as situações, e assim, evitar uma difícil implementação e consequentemente custos elevados.

Numa organização, um produto de IdM além de ser considerado uma tecnologia também se apresenta como um processo de mudança na sua cultura. Desta forma, a garantia do seu sucesso no seio da organização pode ser alcançado através da combinação dos seguintes fatores [7, 8]:

- Escolha pela tecnologia adequada à organização, ou seja, esta deve ser baseada numa análise prévia sobre qual a tecnologia que melhor se adapta à área de negócios da empresa;
- Definição correta dos processos de integração, fluxos de aprovações e *workflows* necessários;
- Determinação de uma cultura interna e transversal a toda a empresa, com divulgação sobre o novo método e procedimentos a adotar de forma a evitar desvios ao estabelecido;
- Gestão de perfis correta baseada numa definição em conformidade com as normas e políticas definidas.

Analisando estes fatores, uma implementação de sucesso verifica-se aquando de uma integração total da empresa com todas as áreas envolvidas. Contudo, hoje em dia existem bastantes tentativas de implementação de produtos IdM disponíveis no mercado, como por exemplo do OIM e que, até ao momento não foram concluídas com sucesso. Apesar de existir atualmente várias aplicações nas organizações que registam os pedidos de acesso, torna-se necessário criar um sistema que centralize essa informação sem quaisquer erros e que permita responder atempadamente às necessidades dos utilizadores e dos administradores de sistemas.

Soluções de gestão de identidades e acessos

Os produtos de gestão de identidades que se destacam atualmente no mercado são: Oracle Identity Management[5] desenvolvido pela empresa Oracle, Microsoft Forefront Identity Manager[6] desenvolvido pela empresa Microsoft e Tivoli Identity[7] Manager desenvolvido pela empresa IBM.

[5] http://www.oracle.com/br/products/middleware/identity-management/overview/index.html

[6] http://www.microsoft.com/pt-br/server-cloud/forefront/identity-manager.aspx

[7] http://www-142.ibm.com/software/products/br/pt/secuidenmana/

a) Oracle Identity Manager

O produto OIM (Oracle Identity Manager) é um componente da solução Oracle IdM (Oracle Identity and Access Management Suite) desenvolvida pela Oracle. A plataforma Oracle Identity Management oferece mecanismos e tecnologias de gestão de identidades e acessos, permitindo às organizações gerir todo o ciclo das identidades dos utilizadores em todos os recursos de rede, e alcançarem uma rápida conformidade com as exigências das leis de TI a que estão sujeitas, protegendo assim as aplicações e dados confidenciais, independentemente de estarem armazenados localmente ou em *cloud* (nuvem) reduzindo também os seus custos operacionais [10].

Figura 2.3 - Oracle Identity Manager

O OIM é considerado uma solução bastante flexível, eficiente e escalável oferecendo uma administração automatizada e centralizada de identidades, um aprovisionamento de identidades, contas e eventos que ocorram no sistema de rede. Desta forma, as organizações poderão usufruir de um sistema automatizado de gestão (criação, atualização e remoção) de utilizadores e de atribuição de diferentes níveis de permissões aos recursos da organização, sem a necessidade de alterar a infraestrutura existente e as políticas de segurança implementadas.

b) Microsoft Forefront Identity Manager

O produto FIM (Microsoft Forefont Identity Manager) é uma solução desenvolvida pela Microsoft para gestão de identidades, credenciais de acesso e perfis baseados em políticas de acesso. O FIM é um produto baseado em linguagens .NET e altamente extensível permitindo os programadores de qualquer empresa produzirem as suas próprias soluções mais personalizadas e expansíveis. Desta forma, as organizações poderão usufruir de um meio para gerir contas de utilizadores e respetivos acessos,

credenciais baseadas em senhas e certificados e políticas baseadas em identidade em ambientes Windows e heterogéneos [17].

Figura 2.4 - Microsoft Forefront Identity Manager

Numa organização, as áreas abrangidas por esta solução são:

- Gestão de utilizadores - consiste em ferramentas para: aprovisionar e desaprovisionar de forma eficiente utilizadores; criar políticas de aprovisionamento e desaprovisionamento de contas, credenciais e recursos; e gerir perfis pelos próprios utilizadores.
- Gestão de credenciais - para administradores e utilizadores finais, possibilitando a gestão de um ciclo de vida das credenciais integrado ao aprovisionamento. Permite também a gestão centralizada de várias credenciais, a sincronização senhas simplificando a operação de *login*, e a possibilidade dos próprios utilizadores redefinirem as suas senhas.
- Gestão de acessos - permite a criação de grupos de acesso, ou seja, com as mesmas propriedades e desta forma evitar a gestão de identidades repetitiva. E assim, associar utilizadores a esses grupos com determinados perfis e políticas de segurança.
- Gestão de políticas - oferece uma estrutura de automação e integração de gestão de identidades para que todos os sistemas da organização utilizem o mesmo conjunto de políticas. Isto é realizado através da criação e auditoria centralizada de políticas associadas a utilizadores e grupos com controlos orientados por níveis, reduzindo o risco de falta de conformidade.

c) Tivoli Identity Manager

O produto TIM (Tivoli Identity Manager) é uma solução de gestão de identidades desenvolvida pela IBM. O TIM constitui um sistema capaz de gerir operações sobre

utilizadores, identidades e privilégios de acesso da infraestrutura de TI de uma organização. Este produto é intuitivo e de fácil instalação em grandes e pequenas organizações, e permite estarem em conformidade com os regulamentos e uma boa gestão dos riscos.

O TIM oferece funcionalidades, tais como:

- Gestão de operações, contas e privilégios de acesso de forma automática acompanhando os respetivos ciclos de vida. Reduzindo assim quaisquer custos adicionais de manutenção.
- Rápida implementação de novos utilizadores e aplicações por políticas e modelos previamente configurados. Reduzindo o tempo de espera no fornecimento de recursos a novos utilizadores.
- Fornece interfaces que ajudam os próprios utilizadores a modificar as suas senhas e/ou informações pessoais. Reduzindo eventuais custos de ajuda ou *helpdesk*.
- Permite a separação de obrigações de forma a aumentar a segurança, associando os requisitos que evitam conflitos de negócios com as funções e políticas de fornecimento que controlam os privilégios de acesso dos utilizadores.
- Atualiza e remove privilégios de acesso que não estão em conformidade por meio de processos periódicos.

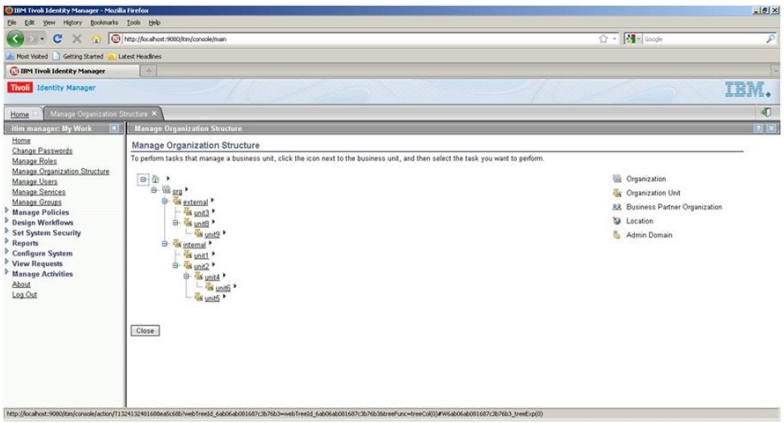

Figura 2.5 - Tivoli Identity Manager

2.2.3 Resolução de entidades

A palavra entidade é algo que existe por si mesmo, não sendo estritamente necessária a sua existência material. No mundo empresarial, uma entidade poderá referir uma pessoa, um departamento, uma equipa, uma parceria, ou outro grupo na qual é

possível realizar operações de negócio. É também muitas vezes referida como uma entidade sem nenhuma forma corporal, referindo-se a uma entidade não física.

Em qualquer organização, os clientes e produtos são uma mais-valia para o seu negócio, e estas armazenam cada vez mais informação ao longo do tempo. No entanto, apenas ter em posse toda a informação não significa que poderemos usá-la de forma eficaz. Se ela não for devidamente integrada, os dados podem na realidade produzir falsas conclusões que resultam em decisões erradas e oportunidades desperdiçadas. Na realidade, é frequente encontrar entidades que têm duas ou mais representações em bases de dados. Registos duplicados não partilham chaves comuns e podem conter erros tornando assim difíceis as tarefas de deteção e combinação de duplicados. Estes erros poderão resultar de incorretas transcrições, informação incompleta ou da falta de formatos *standart* (padrão).

Nas organizações, hoje em dia as bases de dados desempenham um papel importante e central permitindo facilmente realizar operações. Num sistema livre de erros com dados limpos, a apresentação de dados consiste na simples operação de ligação que em temos relacionais, se refere à operação de *joinning* (junção) de várias tabelas. Mas infelizmente, nem sempre existe tais cenários. Além disso, muitas das vezes, os dados não são controlados de forma cuidada para efeitos de qualidade nem definidos de uma forma consistente entre diferentes fontes de dados. Permitindo que a qualidade dos dados seja comprometida por diversos fatores, como por exemplo, a introdução de erros durante a entrada de dados (ex., "Microsft" em vez de "Microsoft"), ausência de regras de integridade (ex., idade = 235) e a utilização de várias convenções para guardar a mesma informação (ex., "Avenida 5 de Outubro, 32" e "32, Av. 5 de Outubro").

Geralmente, durante a integração de dados de diferentes fontes para a implementação de um *datawarehouse*, as organizações devem estar preocupadas para eventuais diferenças sistemáticas e possíveis conflitos. Estes problemas derivam da heterogeneidade dos dados. Por este motivo, é muito importante numa primeira fase fazer uma limpeza aos dados para eliminar estas implicações. Segundo os autores de [11], a heterogeneidade pode ser classificada em dois tipos:

- Heterogeneidade estrutural – existe quando as colunas de uma base de dados estão diferentemente estruturadas em várias bases de dados. Por exemplo, numa base de dados temos a residência guardada num campo "address" e noutra, a mesma informação está guardada em múltiplos campos como "street", "city" e "zip_code".
- Heterogeneidade lexical – existe quando bases de dados têm colunas identicamente estruturadas mas os respetivos valores usam diferentes

representações para referir o mesmo objeto. Por exemplo, campos "StreetAddress" = "44 W. 4th St" e "StreetAddress" = "44 West Fourth Street" em diferentes bases de dados.

Segundos os autores de [5], o processo de Resolução de Entidades consiste num problema de integração de informação, e identificação e relação de diferentes representações da mesma entidade. É também conhecido como Data Deduplication (processo de eliminação de dados duplicados), Record Linkage (processo de combinação de registos), Record Matching, Instance Identification, Duplicate Record Detection, Entity Matching, entre outros mais.

Alguns exemplos de cenários que podem acontecer na realidade:

- Diferentes formas de endereçamento de nomes e endereços de *email* pertencentes a uma mesma pessoa num texto;
- Diferentes páginas *web* com diferentes descrições acerca do mesmo assunto;
- Diferentes fotografias de um mesmo objeto.

Assim, o principal objetivo da resolução de entidades consiste em identificar diferentes ou múltiplos registos na mesma ou em diferentes bases de dados que referem a mesma entidade, mesmo que os registos não sejam idênticos [9]. Por exemplo, dois registos de uma mesma pessoa podem fornecer grafias de nomes diferentes e morada de residência diferentes. Outro cenário onde se poderia aplicar esta técnica seria, por exemplo, no caso de duas empresas que se fundem podem querer combinar os respetivos registos de clientes ou funcionários. Aqui, o mesmo funcionário pode ser representado por vários registos, e portanto registos que combinam devem ser identificados e unificados, naquilo a que se chama num *cluster* (grupo). De seguida é apresentado um exemplo alvo de um procedimento deste tipo.

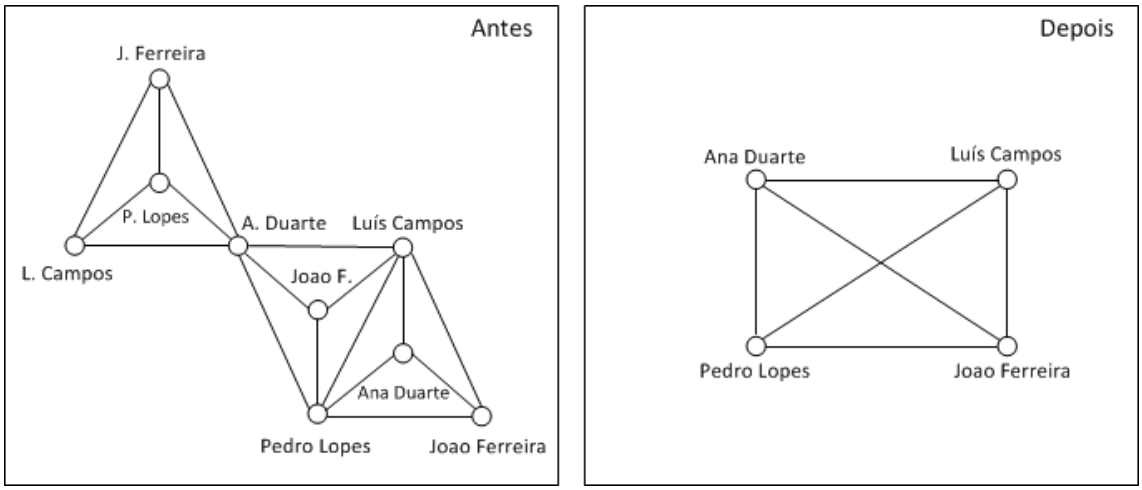

Figura 2.6 - Antes e depois da aplicação desta técnica a uma rede de utilizadores

Como podemos ver, um processo deste tipo tem muitas vezes um elevado custo associado devido à grande quantidade de dados e à necessidade de comparações de computação intensiva entre registos. Por exemplo, o armazenamento de perfis de indivíduos em *websites* sociais pode originar centenas de milhões de registos que necessitam de ser resolvidos. Comparar cada par de registos para estimar a sua "similaridade" pode ser dispendioso, aumentando quando estão envolvidas grandes quantidades de campos comparáveis.

Como foi referido anteriormente, a deteção de duplicados é tipicamente precedida por uma fase de preparação de dados em que as entradas de dados são armazenadas de forma uniforme na base de dados, resolvendo parcialmente o problema de heterogeneidade estrutural. Esta fase é também conhecida pelo termo ETL (Extração, Transformação e Carregamento) e permite o melhoramento da qualidade do fluxo dos dados, tornando estes mais utilizáveis e comparáveis. Tipicamente, a preparação de dados envolve três passos [11]:

- *Parsing* – analisa, identifica e isola elementos de dados individuais nas fontes. Este passo permite corrigir e padronizar os dados, permitindo assim a comparação de elementos individuais, sendo preferível que grandes e complexas *strings* (sequências de carateres).
- *Data Transformation* – simples conversões aplicadas aos dados para satisfazerem os tipos dos seus respetivos domínios. O tipo de conversão foca-se na manipulação de um campo de cada vez. Alguns exemplos são: conversão do campo data de um tipo para outro, renomeação de um nome de um campo para outro e verificação de intervalos numéricos ou datas.
- *Data Standardization* – padronização da informação representada em determinados campos para um formato específico. Esta operação é usada quando a informação pode ser armazenada de formas diferentes em várias fontes e devem ser convertidas para uma representação uniforme antes de iniciar o processo de deteção de duplicados. Este passo é considerado muito importante porque sem ele muitos registos duplicados são classificados erradamente como sendo "não-duplicados" devido ao facto que a informação identificada como comum não pode ser comparada. Por exemplo, a padronização do nome identifica elementos como primeiros, últimos, e nomes intermédios. A formatação de datas e nomes são consideradas tarefas difíceis de padronização, numa base de dados.

Após esta fase de preparação, os dados são normalmente armazenados em tabelas com campos possíveis de serem comparados. De seguida, o próximo passo é identificar que campos deverão ser alvo de comparação, de forma a evitar comparações de campos como "FirstName" e "Age".

Os métodos existentes para resolver entidades incluem a comparação de valores de variáveis, como nomes, endereços, datas e outros associados às entidades em cada par de registos, para verificar se um determinado par corresponde à mesma entidade. De todo o conjunto de variáveis, uma das mais importantes para identificar uma entidade é o nome [18]. Pois muitas pessoas podem ter o mesmo nome, e os programas de *software* devem levar em conta informação adicional além do nome para determinar se dois registos idênticos na verdade se referem à mesma pessoa. Desta forma, o processo de resolução de entidades constitui um problema difícil de ser resolvido em muitos casos porque não existe informação suficiente para levar a uma determinação precisa. Pois, os nomes podem ser apresentados em vários formatos utilizando abreviaturas ou variações na sua escrita, com erros de digitação, além de poderem mudar ao longo do tempo, por exemplo, devido ao casamento, divórcio, entre outros. Portanto, a deteção de duplicados tipicamente depende de técnicas de comparação de *strings* para lidar com este problema, sendo estas um elemento chave para determinar se um par de registos representa ou não a mesma entidade.

Os métodos tradicionais de pesquisa em bases de dados utilizam uma pesquisa por coincidência exata [19]. Numa base de dados de locais, por exemplo, "Campo Grande" e "C. Grande" são considerados locais diferentes. Desta forma, a obtenção de informação por meio de consultas exatas pode não ser suficiente para estes tipos de casos. Por isto, torna-se necessário utilizar métodos de pesquisa mais sofisticados que permita consultas aproximadas, resultando em consultas por similaridade.

Adicionalmente, funções de similaridade podem ser usadas em outras áreas como na Medicina para análise de ADN, na deteção de plágio, na pesquisa de textos quando não se sabe a grafia correta, cruzamento de dados de bases de dados, entre outros.

Técnicas de combinação de campos

Existem várias técnicas usadas para a combinação individual de campos de um registo, e cada técnica funciona bem para um particular tipo de erros. De modo geral, estas funções para determinar o quão parecido são duas *strings* retornam um valor entre 0 e 1, sendo o valor 1 obtido quando são totalmente idênticas. O grande número de métricas existentes reflete o grande número de erros que podem surgir nos dados na realidade. São apresentados de seguida quatro grupos de métricas diferentes [11, 18, 19].

Métricas de similaridade baseada em carateres

As funções de similaridade baseadas em carateres usam uma distância de edição para comparar todos os carateres de uma *string* com os carateres de outra, em que a distância de edição diz respeito ao número de operações necessárias para transformar

uma *string* noutra. Estas métricas permitem lidar com erros tipográficos. De seguida, são apresentadas algumas métricas deste tipo, sendo a métrica de Jaro mais detalhada uma vez que foi a escolhida para este projeto:

a) Edit Distance

São métricas que funcionam bem para capturar erros tipográficos mas no entanto não são aconselhadas para *strings* truncadas ou encurtadas, como por exemplo, "John R. Smith" e "Jonathan Richard Smith".

O valor de Edit Distance entre duas *strings* A e B consiste no número mínimo de passos de edição de carateres singulares, necessários para transformar a *string* A na *string* B. Os tipos de operações de edição são: inserir, apagar, e substituir um carater por outro carater diferente. Esta versão é também designada por Levenshtein distance.

b) Affine Gap Distance

São métricas que funcionam bem para *strings* truncadas ou encurtadas, ao contrário da métrica Edit distance. Esta métrica introduz duas novas operações de edição: *opening gap* e *extend gap*.

c) Smith-Waterman Distance

É uma extensão das métricas Edit Distance e Affine Gap Distance em que incompatibilidades no início e no fim da *string* têm menor custo que incompatibilidades no meio. Esta métrica funciona bem na combinação de *sub-strings*, por exemplo, "Prof. John R. Smith, University of Calgary" e "John R. Smith, Prof" combinariam numa curta distância desde que os prefixos e sufixos sejam ignorados.

d) Jaro Distance Metric

Jaro desenvolveu um comparador que leva em conta o número de carateres e os tipos de erros que ocorrem frequentemente com variáveis alfanuméricas. É uma métrica de comparação de *strings* usada principalmente para comparação de pequenas *strings* como primeiros e últimos nomes.

O comparador de Jaro é dado pela seguinte fórmula:

$$dj = \ 1/3 \times (c/d + c/r + (c - t)/c)$$

Onde:

dj : comparador de Jaro
c : número de carateres em comum nas duas *strings*
d : número de carateres da primeira *string*
r : número de carateres da segunda s*tring*
t : número de transposições de carateres

Se *c* = 0 então *dj* = 0

Dois caracteres são considerados comuns se estão em ambas as *strings* sendo que a distância entre as posições do caracter em ambas as *strings* pode ser no máximo a metade do comprimento da menor delas. Transposições são caracteres que estão posicionados em locais diferentes nas *strings* em comparação.

Exemplo: dada as duas seguintes *strings*, o valor da métrica de Jaro é dado pela seguinte fórmula:

Jaro ("Thiago Silva", "Tiago Silvio") = (1/3) x (10/12 + 10/12 + (10-0) / 10) = 32/36 =~ 0.88

Posteriormente, Winkler [Winkler 1994] modificou a métrica de Jaro para dar um maior peso ao resultado de Jaro com o objetivo de valorizar strings que tenham um prefixo em comum. O valor deste comparador é dado por:

$$dw = dj + i \times (1 - dj)/10$$

Onde:

dw : comparador de Winkler
i : número de carateres (ou tamanho) do prefixo mais longo
dj : comparador de Jaro

Esta métrica adiciona um maior peso ao resultado de Jaro com o objetivo de valorizar *strings* que tenham um prefixo em comum.

Exemplo: dadas as duas *strings* anteriores, o valor da métrica de Winkler é dado pela seguinte fórmula:

JaroWinkler ("Thiago Silva", "Tiago Silvio") = (32/36) + (1/10) x (4/36) = 0.9

Segundo os autores de [21] e Yancey [16], com base nos testes e análises realizadas a um conjunto de dados, permitiram concluir que o método Jaro e a sua extensão Jaro-Winkler possuem um bom desempenho e são substancialmente mais eficientes, sendo a métrica Jaro-Winkler a melhor pois é baseada no número e ordem de carateres comuns entre duas *strings*. Estas duas métricas servem para integrar bases de dados heterogéneas e a combinação de meta dados, sendo particularmente boa para deteção de erros ortográficos.

e) Q-Gram Distance

Q-grams são *substrings* de poucos carateres (ou sequências de *n* palavras) de comprimento *q*. O objetivo consiste em usar *q-grams* como uma base para a combinação de *strings*. Quando duas *strings* são similares, elas têm um grande número de *q-grams*. Dada uma *string* A, os seus *q-grams* são obtidos por "deslizamento" da janela de comprimento *q* sobre os carateres de A.

Existe uma extensão deste algoritmo, designado por Positional Q-Grams que adicionalmente regista a posição do *q-gram* na *string*, sendo esta variação usada para localizar eficientemente *strings* similares numa base de dados.

Métricas de similaridade baseada em tokens

As métricas de similaridade baseadas em caracteres funcionam bem quando existem erros tipográficos, mas no entanto, é frequente os casos em que convenções tipográficas levam à reorganização de palavras, como por exemplo, "John Smith" e "Smith, John". Neste exemplo, as métricas baseadas em carateres falham ao capturar a similaridade das entidades, no entanto, as baseadas em *tokens* resolvem este problema.

Estas funções dividem as *strings* em palavras separadas por espaços e usam essas palavras como elementos das suas métricas. Assim, os *tokens* de uma *string* são comparados com os *tokens* de outra e apenas são considerados similares se forem idênticos [19]. Algumas métricas existentes são:

a) Atomic Strings

São utilizadas na combinação de campos de texto. Uma *string* atómica consiste numa sequência de carateres alfanuméricos delimitados por carateres de pontuação. Duas *strings* atómicas combinam se são iguais ou se uma é o prefixo da outra. Baseado neste algoritmo, a similaridade de dois campos é o número das suas *strings* atómicas correspondentes dividido pelo número médio de *strings* atómicas.

b) WHIRL

Consiste num sistema que adota a partir da recuperação de informação a similaridade do cosseno combinado com o valor tf-idf (ou seja, a medida usada para calcular a frequência ou importância dos termos dentro de um conjunto de documentos) para computar a similaridade de dois campos. A métrica de similaridade do cosseno funciona bem para uma grande variedade de entradas sendo insensível à localização de palavras, permitindo assim movimentos e trocas de palavras, como por exemplo, "John Smith" é equivalente a "Smith, John". Esta métrica não captura erros de ortografia de palavras, especialmente se elas afetam muitas palavras nas *strings*, por exemplo, "Compter Science Department" e "Deprtment of Computer Scence" terão zero de similaridade usando esta métrica.

c) Q-Grams com tf-idf

Consiste num sistema baseado na extensão do sistema WHIRL para controlar erros de ortografia usando *q-grams*, em vez de palavras. Nesta configuração, um erro ortográfico afeta minimamente o conjunto de *q-grams* comuns de duas *strings*, por exemplo, "Gteway Communcations" e "Communications Gateway" têm um alto valor

de similaridade usando esta métrica, desprezando o movimento do bloco e os erros ortográficos em ambas as palavras. No entanto, "Gateway Communications" combina com alto valor de similaridade "Communications Gateway International" desde que o *q-gram* da palavra "International" apareça frequentemente na relação e tenha um baixo peso.

Métricas de similaridade fonéticas

As métricas de similaridade baseada em carateres e baseada em *token*s focam-se na representação baseada em *strings* de registos da base dados. Contudo, as *strings* podem ser foneticamente similares ainda que elas não sejam similares num carater ou ao nível de um *token*. Por exemplo, a palavra "Kageonne" é foneticamente similar a "Cajun" assim como "Smith" e "Smithe", apesar de que as representações das *strings* serem muito diferentes. As métricas existentes para tratar estes tais problemas e combinar estas *strings* são: Soundex, NYSIIS (New York State Identification and Intelligence), ONCA (Oxford Name Compression Algorithm), e Metaphone and Double Metaphone.

A Fonética é o ramo da Linguística que se preocupa com a parte significante do som linguístico e não com o seu conteúdo.

Métricas de similaridade numérica

Tipicamente, os números são tratados como *strings*, e por sua vez são comparados usando as métricas descritas anteriormente.

Técnicas de combinação de registos

Anteriormente foram analisados os métodos que são utilizados para combinar campos individuais de um registo. Ao nível da combinação de registos, estes são constituídos por vários campos tornando o problema de deteção ainda mais complicado. Os métodos usados para combinar registos podem ser divididos em duas abordagens [11, 18]:

- Abordagem probabilística que depende de Data Training (dados de treino) para "aprender" como combinar registos. Podem ser baseados na teoria de ligação de registos probabilística clássica, como a desenvolvida por Ivan Fellegi e Sunter [11] ou baseada em técnicas de aprendizagem de máquina supervisionada.
- Abordagem determinística que depende do conhecimento do domínio ou métricas de distância genéricas para combinar registos. Podem ser utilizadas regras ou identificadores únicos que permitem determinar quando dois pares de registos referem a mesma entidade ou não.

Os autores de [21] associam a combinação de entidades a um problema de classificação, onde o principal objetivo é classificar pares de entidades como pares

verdadeiros caso pertençam à mesma entidade, caso contrário, são pares falsos. A notação usada nestas técnicas é a seguinte: A e B designam as tabelas que queremos fazer a combinação, e estas têm *n* campos comparáveis. Cada par de registos de A e B é atribuído a uma de duas classes, M e U. A classe M contem pares de registos que representam a mesma entidade (*match*) e a classe U contem pares de registos que representam duas entidades diferentes (*nonmatch*). Cada par é representado por um vetor *v* com *c* componentes que correspondem a *x* campos comparáveis.

Alguns métodos seguindo estas abordagens são [11]:

Técnicas baseadas em distância

Permitem a definição de uma métrica de distância em que é necessário um *threshold* (um limite) apropriado para a combinação sendo assim possível combinar registos similares sem o uso de dados de treino. Uma simples abordagem consiste em medir a distância entre campos individuais, usando a métrica de distância apropriada para cada campo, e por fim, calcular a distância ponderada entre os registos. Mas neste caso temos o problema de calcular os pesos e a configuração global tornando-se muito próxima à configuração probabilística.

Uma abordagem alternativa é criar uma métrica de distância que é baseada numa *ranked list* (lista de classificação). A ideia base consiste em comparar apenas um campo do registo permitindo ao algoritmo de combinação facilmente encontrar as melhores combinações e classifica-las de acordo com a sua similaridade. Por aplicação do mesmo princípio para todos os campos, conseguimos obter para cada registo *n ranked lists*, uma para cada campo. De seguida, o objetivo é criar um *rank* de registos que têm a distância mínima de classificação agregada quando comparado com todas as *n* listas.

Técnicas baseadas em regras

São um caso particular das abordagens baseadas na distância. Consiste no uso de regras para definir se dois registos são o mesmo ou não. Estas técnicas permitem que em casos que não exista uma chave global, se utilize regras desenvolvidas por especialistas para obter um conjunto de atributos que em conjunto funcionam como uma "chave" para cada registo. Por exemplo, temos as seguintes regras:

```
    IF idade < 22 THEN status = undergraduate ELSE status = graduate

    IF distanceFromHome > 10 THEN transportation = car ELSE transportation =
bicycle
```

Usando tais regras pode-se agrupar múltiplos registos que representam a mesma entidade. O resultado das regras deve ser sempre o correto. Por isso, as regras não

devem ser heuristicamente definidas mas deverão refletir verdades absolutas e servir como dependências funcionais.

Ao utilizar regras, é possível, por exemplo, se duas pessoas têm nomes de grafia similar e têm o mesmo endereço, podemos inferir que elas são a mesma pessoa. Especificando tal inferência numa teoria equacional requer uma linguagem de regras declarativas. Por exemplo, a seguinte regra exemplifica a combinação de registos de uma base de dados de empregados:

```
FORALL (r1,r2) in EMPLOYEE

  IF r1.name is_similar to r2.name AND r1.address = r2.address

THEN r1 matches r2
```

De notar que o valor de "is_similar" é calculado por uma técnica de comparação de *strings* e "matches" significa declarar que esses dois registos combinam e portanto representam a mesma pessoa.

Existem ainda outras técnicas que envolvem cálculos probabilísticos em que os pares de registos são representados não por pares de *strings* mas por *match features* (vetores de características) como nomes e categorias para variáveis de bases de dados [21].

Concluindo, um processo de Resolução de Entidades tem muitas vezes um elevado custo associado devido à grande quantidade de dados e à necessidade de comparações de computação intensiva de registos. A eficiência do processo de deteção de registos duplicados pode ser aumentada reduzindo o número de comparações de registos ou melhorando a eficiência do algoritmo de comparação de registos, como poderemos ver o desenho da solução desenvolvida no capítulo 4. Por fim, podemos ver algumas ferramentas especializadas no processo de deteção de registos duplicados.

Ferramentas para deteção de duplicados

Muitos fornecedores de bases de dados como a Orable, IBM e Microsoft não providenciam ferramentas suficientes para detetar registos duplicados mas existem os seguintes produtos para o efeito, sendo alguns de utilização gratuita.

a) FEBRL (Freely Extensible Biomedical Record Linkage)[8]

   É um sistema *open-source* de ferramentas de limpeza de dados, e é constituído por dois componentes: um que realiza a padronização dos dados e outro que trata da deteção de duplicados. No processo de deteção, o sistema tem uma grande

[8] http://sourceforge.net/projects/febrl/

variedade de métricas de similaridade de *strings*, tais como Edit Distance, Jaro e Q-Gram Distance. Ainda suporta similaridade fonética para detetar nomes.

b) TAILOR

É um conjunto de ferramentas de combinação de registos flexível que permite aos utilizadores definirem diferentes métodos de deteção sobre os dados.

c) WHIRL (Word-based Information Representation Language)[9]

É um sistema de deteção de registos duplicados disponível de forma gratuita para uso académico e investigação. O sistema utiliza a métrica de similaridade baseada em *tokens* Q-Grams com tf-idf para identificar *strings* similares.

d) BigMatch

É uma ferramenta desenvolvida para identificar potenciais combinações de registos e é bastante escalável para grandes conjuntos de dados. Sendo o principal objetivo desta ferramenta não executar uma deteção de duplicados precisa mas sim gerar um conjunto de pares candidatos que deverão ser processados por algoritmos de deteção de duplicados sofisticados.

2.3 Análise de números de identificação

Uma vez que este projeto vai lidar muito frequentemente com a desambiguação de números identificativos, como por exemplo, perceber qual o tipo de documento de identificação se trata um determinado valor de "user_id" ou "login" de uma conta, tornou-se necessário realizar previamente um estudo aos vários tipos de documentos de identificação utilizados internamente pela Portugal Telecom a fim de compreender a sua estrutura e o seu peso representativo [22].

Esta análise também contribuiu para o desenvolvimento do módulo de validação de dados, caracterizado no subcapítulo 4.5. Neste contexto é importante conhecer as regras de construção dos seguintes números:

- Número de Bilhete de Identidade;
- Número de Contribuinte;
- Número de Passaporte;
- Número de Autorização de Residência;
- Número de Colaborador.

[9] http://www.cs.cmu.edu/~wcohen/whirl/

O número de Passaporte consiste num documento de identidade emitido por um governo nacional que atesta formalmente o portador como nacional de um estado em particular e é constituído por números e letras. Estes números são designados por uma autoridade emissora e não seguem um formato genérico. Desta forma, não é possível proceder à validação destes números, pois são atribuídos conforme o país emissor. Por outro lado, o número de Autorização de Residência consiste num documento que é emitido sobre a forma de um título de residência e que permite aos cidadãos estrangeiros permanecer em Portugal. Não existe também um formato genérico para a construção destes números, não sendo possível também efetuar a validação dos mesmos.

De seguida, são apresentados os números de documentos mais importantes e mais utilizados pela organização, e possíveis de serem validados.

2.3.1 Número de bilhete de identidade

O número de Bilhete de Identidade Português consiste num documento nacional de identificação civil em Portugal, que foi substituído por um novo documento de identificação, designado por, Cartão de Cidadão. O Cartão de Cidadão, ao contrário do Bilhete de Identidade que apenas armazena o seu número de identificação, este possui outros documentos de identificação (como, cartões da Segurança Social, Serviço Nacional de Saúde, Eleitor e Número de Identificação Fiscal), e começou a ser emitido a partir do ano de 2008.

Cada Bilhete de Identidade tem um número de identificação (na parte de trás no canto superior esquerdo) e mantém-se inalterável ao longo da vida do indivíduo. Devido à elevada importância deste documento, é fundamental que a transmissão do respetivo número decorra sem erros, sendo este o principal objetivo do algarismo de controlo, que se encontra à direita do número de BI, num pequeno quadrado. Assim o algarismo de controlo permite verificar se o número de BI com aquele algarismo é válido ou não, e caso não seja, poderá ter havido um engano a escrever o número ou o BI em questão é falso.

Atualmente, estes algarismos de controlo são utilizados noutros documentos que necessitem de uma maior segurança, como por exemplo, cartões de crédito, cheques, Via Verde, códigos de barras (UPC-EAN), livros (ISBN), entre outros. De forma resumida, o objetivo dos algarismos de controlo é permitir uma deteção imediata quando alguém comete um erro ao inserir um destes números em qualquer sistema informático.

Processo de validação

Em Portugal, o número de Bilhete de Identidade é composto por 9 dígitos, em que o 9º algarismo corresponde ao dígito de controlo. Mas, tipicamente apenas são fornecidos os primeiros 8 dígitos. O algarismo de controlo do BI é um algarismo que é calculado a partir do número de identificação da seguinte forma:

$$9x1 + 8x2 + 7x3 + 6x4 + 5x5 + 4x6 + 3x7 + 2x8 + C = 0(mod\ 11)$$

Onde *C* é o algarismo de controlo, *x1* é o 1º algarismo do número de BI, *x2* é o 2º, *x3* é o 3º e assim sucessivamente. Aos números de BI que tiverem menos de oito algarismos, deverão ser acrescentados os zeros necessários à esquerda do número, até este perfazer oito algarismos. Por exemplo, o número 123456 deve ser encarado como 00123456. De qualquer modo, *C* é o número que somado a (9*x1* + 8*x2* + ... + 2*x8*) dá origem a um múltiplo de onze, ou seja, o número *C* é escolhido de modo a (9*x1* + 8*x2* + ... + 2*x8* + *C*) dê resto zero quando dividido por onze.

Uma vez que estamos a considerar a divisão por onze, o algarismo de controlo pode tomar o valor 0 (quando a divisão de 9*x1* + 8*x2* + ... + 2*x8* por onze tiver resto zero), o valor 1 (quando a divisão tiver resto 10), o valor 2 (quando a divisão tiver resto 9) e assim sucessivamente até chegarmos ao número 10 (que é atingido quando a divisão tiver resto 1).

No caso de querermos calcular o algarismo de controlo a partir do número de BI, por exemplo, "1569448" (que no fundo seria 01569448), começaremos por multiplicar sucessivamente a partir das unidades cada um dos algarismos do número por 2, por 3, por 4, …, 9 e adicionar os produtos obtidos, da seguinte forma:

= 2 x **8** + 3 x **4** + 4 x **4** + 5 x **9** + 6 x **6** + 7 x **5** + 8 x **1** + 9 x **0**
= 16 + 12 + 16 + 45 + 36 + 35 + 8 + 0
= 168

Em seguida dividimos a soma obtida por 11:

168	|_11__
58	15
3	

Se somarmos ao número 168 o dígito de controlo, a soma fica divisível por 11, então o algarismo de controlo obtém-se fazendo a diferença entre 11 e o resto obtido, que neste caso é 3.

11 - 3 = 8 (dígito de controlo)

Assim, nos nossos documentos de BI o algarismo de controlo é um número inteiro entre 0 a 9. Quando aparece um 0 este pode na realidade significar zero (0) ou dez (10).

Número de cartão de cidadão português

O número de documento do Cartão de Cidadão apresenta 4 carateres adicionais, quando comparado com o número de Bilhete de Identidade. As duas primeiras parcelas de letras correspondem ao antigo número de Bilhete de Identidade. O formato deste documento é o seguinte:

DDDDDDDD C AAT (ex: 10318369 8 ZZ6)

Onde:

- D – Significa o número de Identificação Civil do titular, constituído por algarismos de 0 a 9.
- C – *Check Digit* do número de Identificação Civil, que permite detetar erros de transcrição deste número.
- A – Representa o número da emissão do cartão para determinada pessoa. A ordem por que aparecem permite aferir o número de Cartões emitidos para um cidadão, por exemplo, "ZZ" corresponde à 1ª emissão, "ZY" corresponde à 2ª emissão, e assim sucessivamente.
- T – *Check Digit* do número de documento que permite detetar erros na transcrição do número de documento completo.

2.3.2 Número de contribuinte

O número de contribuinte, ou também conhecido por, número de Identificação Fiscal (NIF), que em Portugal, é um número atribuído pelo Ministério das Finanças e da Administração Pública para identificar uma entidade fiscal ou um contribuinte, por exemplo, em declarações de IRS ou IRC. No caso de empresas, este número é conhecido por número de Identificação de Pessoa Coletiva (NIPC). De seguida veremos a sua estrutura bem como a sua validação.

Este número é constituído por 9 dígitos. Em que o 1º dígito, sendo aquele que está mais esquerda, tem os seguintes valores possíveis:

- 1 ou 2, se pessoa singular;
- 5, se pessoa coletiva;
- 6, se coletiva pública;
- 8, se empresário em nome individual;

- 9, se pessoa coletiva irregular ou número temporário.

O 9º e último dígito, também designado por, dígito de controlo, possui um papel importante no processo de verificação de validade e autenticidade de um valor numérico, evitando dessa forma erros de transmissão ou digitação, e fraudes. Tal como o Bilhete Identidade, consiste num algarismo adicionado ao número original e calculado a partir deste através de um algoritmo.

Processo de validação

Para validar este documento existem algumas regras que devem ser obedecidas, nomeadamente:

- O número tem de ter 9 dígitos.
- O primeiro dígito tem que ser 1,2,5,6,8 ou 9.
- A soma de controlo é calculada por:

$$9 \times d1 + 8 \times d2 + 7 \times d3 + 6 \times d4 + 5 \times d5 + 4 \times d6 + 3 \times d7 + 2 \times d8 + 1 \times d9$$

Ou, por outras palavras, $(10 - \boldsymbol{i}) \times d\boldsymbol{i}$, em que $\boldsymbol{i}$ vai de 1 a 9.

O resultado desta soma de multiplicações tem que ser múltipla de 11 ou congruente com o módulo 11 (isto é, dê resto 0 quando dividida por 11). Sendo que *d1* é o dígito mais à esquerda e *d9* o dígito mais à direita, neste caso o dígito de controlo.

- Em alguns casos, *d9* precisava de ter o valor 10, para que a soma de controlo fosse divisível por 11, e neste caso *d9* terá o valor 0.

A única diferença entre o número de Contribuinte e o nnmero de Bilhete de Identidade reside no facto de o algarismo de controlo no BI ser indicado num quadrado à parte, enquanto no NIF é o último algarismo do próprio número.

2.3.3 Número de colaborador

O número de colaborador ou também conhecido por número mecanográfico corresponde a um conjunto total de 8 dígitos, e, muitas vezes, este número encontra-se ligado a um prefixo "PT".

2.4 Ferramentas

Neste capítulo são apresentados os papéis desempenhados pelas tecnologias e ferramentas de *software* utilizadas no desenvolvimento, bem como os motivos que levaram a escolher tais tecnologias e o seu enquadramento com o projeto. Para algumas dessas tecnologias foi necessário previamente realizar uma preparação inicial devido ao

insuficiente conhecimento para trabalhar com elas. Todas as tecnologias e ferramentas utilizadas são 100% *open source*.

2.4.1 Ambiente de desenvolvimento

O ambiente de trabalho escolhido foi o sistema operativo Microsoft Windows[10] para o desenvolvimento do projeto, nomeadamente a versão Windows 7. Além disto, e uma vez que os servidores da rede corporativa utilizam o sistema operativo Linux[11], foi necessário instalar também este sistema operativo para uma adequada e correta interação com estas máquinas. Para isto, foi utilizada uma máquina virtual recorrendo ao programa de *software* Oracle VM VirtualBox[12], e de seguida proceder à criação de uma máquina virtual com o sistema operativo Linux.

Para complementar esta fase de preparação do ambiente de desenvolvimento, foi utilizado em complemento um programa de *software* de gestão de projetos, designado por Redmine[13], permitindo a representação visual dos projetos, respetivos *deadlines*, calendário, gráficos de Gantt, entre outras funcionalidades essenciais, possibilitando ainda a utilização integrada de um sistema de controlo de versões de código desenvolvido, designado por Git[14].

2.4.2 Fase algorítmica

Na fase algorítmica, a linguagem de programação utilizada no desenvolvimento de qualquer procedimento ou função do projeto foi a linguagem Ruby[15], e a maior parte do trabalho incidiu sobre esta fase.

Ruby é considerada uma linguagem de script, dinâmica, *open source* e foca-se na sua simplicidade e produtividade, suporta programação funcional, é orientada a objetos e imperativa. Foi inspirada em linguagens como Python[16] e Perl[17], e é atualmente uma das linguagens de programação mais popular do mundo. Hoje em dia, existem várias implementações alternativas desta linguagem, entre elas temos JRuby, IronRuby,

[10] http://windows.microsoft.com/pt-PT/windows/home

[11] http://www.linux.org/

[12] https://www.virtualbox.org/

[13] http://www.redmine.org/

[14] http://git-scm.com/

[15] http://www.ruby-lang.org/pt/

[16] http://www.python.org/

[17] http://www.perl.org/

MacRuby e HotRub. Neste projeto será utilizada em concreto a variante JRuby[18] v1.7.3 que não é nada mais do que uma implementação da linguagem Ruby para a plataforma Java, sendo executada através da máquina virtual do Java[19]. Esta escolha foi necessária para a utilização da métrica de similaridade Jaro Winkler Distance recorrendo à classe Java "JaroWinklerDistance" da biblioteca externa"org.apache.lucene.search.spell", pertencente ao Apache Lucene[20].

Sendo o Pulso uma plataforma completamente *open source* convencionou-se que todos os módulos que viriam a incorporar esta plataforma fosse LAMP/R (isto é, Linux, Apache, MySQL, PHP/Perl e Ruby), e desta forma, é feita grande pressão para a utilização por parte de todos os colaboradores de um conjunto uniforme de ferramentas, dando especial ênfase a linguagens como o Ruby dada a sua elegância e atualidade em termos dos mecanismos que implementa e dos paradigmas de programação que suporta (como por exemplo, objetos, *mixins*, *aspects*, iteradores, *threads*, meta-programação, entre outros.) e a estética visual e intelectual dos seus programas.

A ferramenta de desenvolvimento utilizada para produção de aplicações deste tipo foi o Aptana Studio 3[21].

2.4.3 Armazenamento da informação

A *datawarehouse* a desenvolver tem como objetivo armazenar informação consolidada relativa às atividades da Portugal Telecom. Desta forma, guardar e disponibilizar os resultados devidamente processados e tratados sem quaisquer erros e corretamente validados e categorizados através de diversos mecanismos, foi necessário utilizar uma base de dados relacional para posteriormente ser possível representar graficamente o *datawarehouse.* Este tipo de base de dados possibilita a produção de relatórios de auditoria e outras operações úteis sobre os dados.

A base de dados relacional escolhida foi o MySQL[22]. Esta opção deveu-se ao facto não só da plataforma Pulso ser implementada segundo o padrão LAMP/R mas também de ser tratar de uma ferramenta *open source*, não envolvendo quaisquer custos associados a licenças e manutenção, como por exemplo, Microsoft SQL Server[23] e

[18] http://jruby.org/

[19] http://www.java.com

[20] http://lucene.apache.org/core/

[21] http://www.aptana.com/

[22] http://www.mysql.com/

[23] http://www.microsoft.com/en-us/sqlserver/default.aspx

Oracle SQL[24]. O MySQL consiste num sistema de gestão de base de dados que utiliza a linguagem SQL (Structured Query Language) com uma interface gráfica incorporada. As principais características desta ferramenta são:

- Portabilidade, suportando qualquer plataforma;
- Compatibilidade, através da existência de *drivers* e módulos de interface para várias linguagens de programação;
- Excelente desempenho e estabilidade;
- Facilidade de utilização;
- Suporta controlo transacional e *triggers*;
- Utilização de poucos recursos de *hardware*.

[24] http://www.oracle.com/us/products/database/overview/index.html

Capítulo 3

Arquitetura

Neste capítulo é apresentada toda a base que suporta o trabalho realizado, começando por uma descrição do funcionamento geral do sistema. De seguida, é feita a definição do modelo de dados para o armazenamento dos dados processados apresentando de forma detalhada as bases de dados envolvidas. Por último, descreve-se a arquitetura do sistema sob várias perspetivas e respetivos componentes interrelacionados.

3.1 Descrição do sistema

O projeto encontra-se inserido num sistema dividido em duas grandes camadas: Gestão de Identidades e Monitorização de acessos. A figura 3.1 demonstra o enquadramento do projeto e identifica claramente camadas, diferenciando também os mundos físico e lógico, e as fontes de dados envolvidas. Ao nível da gestão de identidades (camada azul) existem três categorias de bases de dados:

A - Base de dados com informação dos cartões de colaboradores usados no controlo de entrada/saída de edifícios;

B - Base de dados da AD (Active Directory*)* com informação sobre contas de acesso à rede interna;

C - Bases de dados com informação sobre contas aplicacionais, de bases de dados e de sistema.

Esta camada é responsável por descobrir a verdadeira identidade ou personalidade mestre de uma *user account* e ao mesmo tempo combinar num único registo todos os outros considerados duplicados. Permite também atribuir um grau de qualidade à informação sintetizada, quer seja a nível de estrutura quer a nível de validade. E por fim, caracterizar o perfil da identidade por de trás de uma conta.

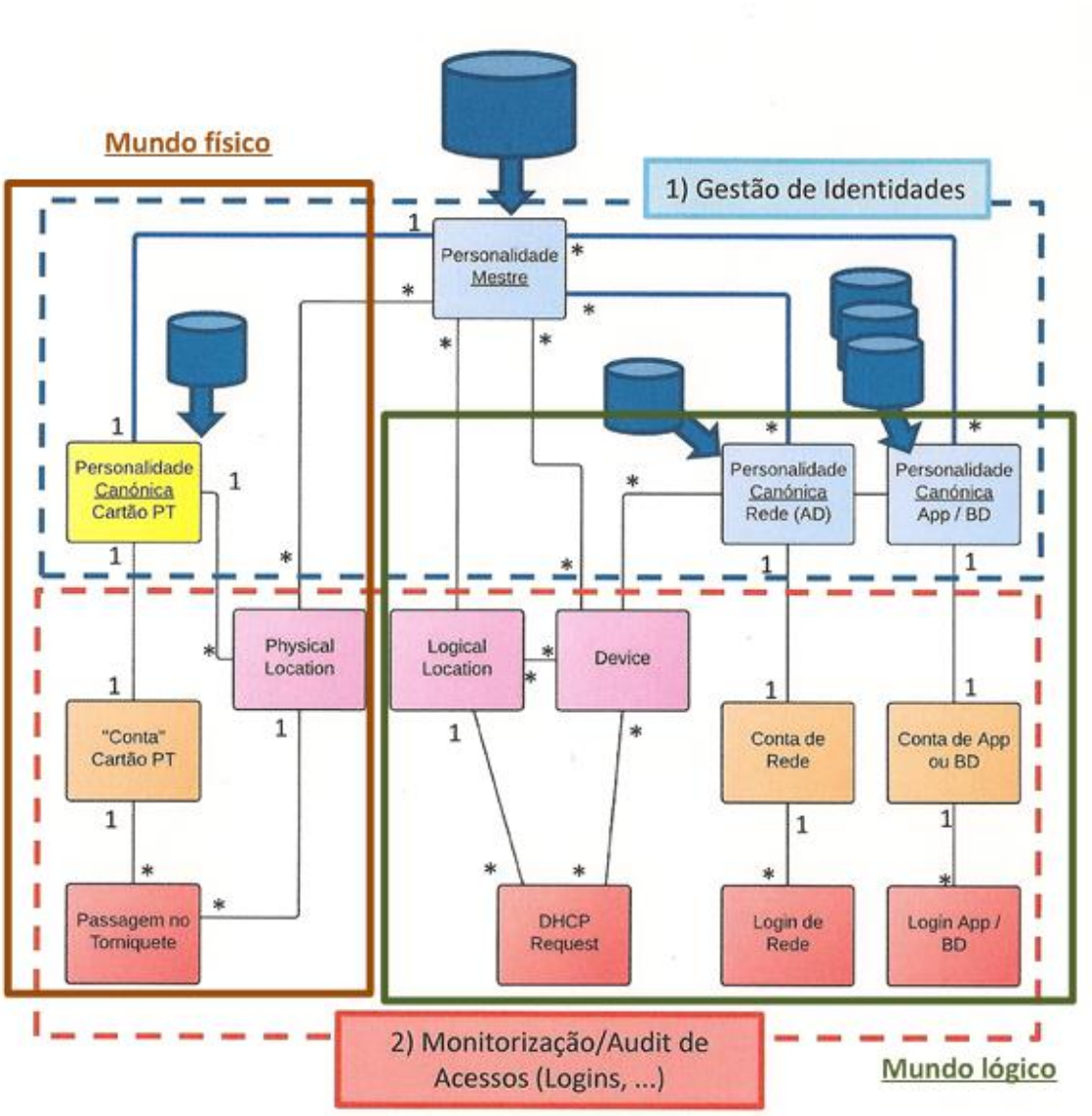

Figura 3.1 – Enquadramento do projeto

Por outro lado, a monitorização de acessos (camada vermelha) pretende complementar a camada anterior, sendo responsável pela captura dos diversos eventos lógicos e físicos, e pela atribuição destes às identidades sintetizadas. Os eventos podem ser simples *logins* (acessos) a aplicações como passagens de torniquete. Esta camada de captura de eventos é responsável por outra equipa da Portugal Telecom, que em paralelo com o desenvolvimento deste projeto utiliza a informação sintetizada e processada para fazer a associação de um evento capturado a uma identidade conhecida.

Segundo os objetivos descritos no subcapítulo 1.2, o sistema a desenvolver consiste na disponibilização de uma *datawarehouse* central sendo focada principalmente no processo de Resolução de Entidades ao nível da Gestão de Identidades. Com isto, pretende-se gerir e determinar a identidade (personalidade mestre) a que pertence na realidade cada *user account* (personalidade operacional), produzindo assim um único

registo central e canónico, caracterizado por um conjunto de informações e anotações importantes para um analista de segurança, excluindo quaisquer incoerências.

3.2 Modelo de dados

De modo a suportar o sistema foi desenhado um modelo de dados capaz de guardar toda a informação e respetivas relações, o qual foi sofrendo alterações ao longo do tempo conforme as necessidades. Inicialmente procedeu-se à modelação de um diagrama EA (Entidade/Associação) para representar o modelo de dados do sistema com alto nível de abstração. Nesta representação podemos ver as entidades envolvidas assim como as suas dependências.

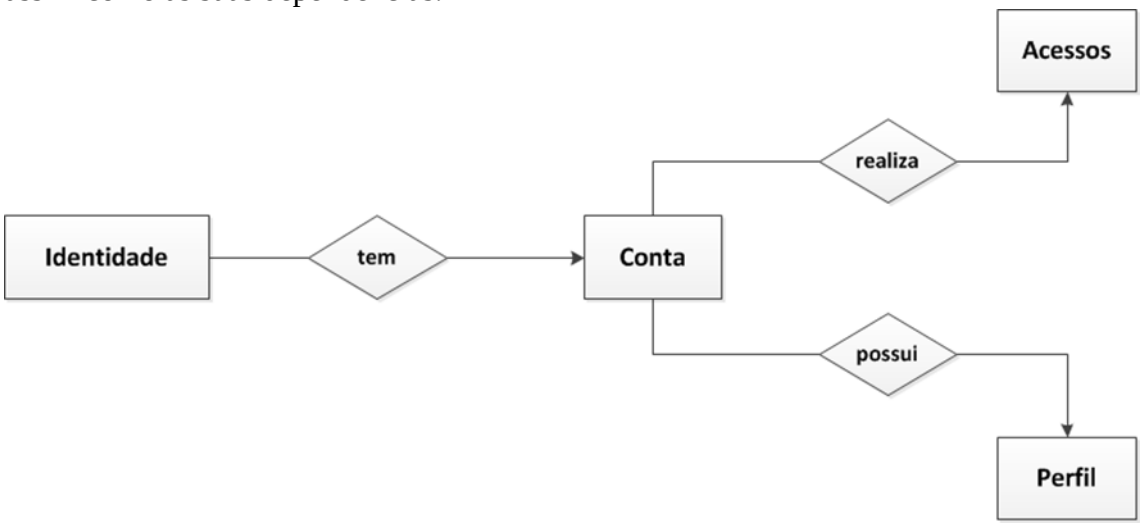

Figura 3.2 - Diagrama EA do sistema de alto nível

O sistema desenvolvido usufrui de três distintas bases de dados relacionais com diferentes propósitos, como poderemos ver de seguida.

3.2.1 Base de dados – PiasaSources

A base de dados PiasaSources é usada para fornecimento de informação *raw* (dados em bruto) para o *input* ou entrada de dados do projeto. Esta contém as fontes de dados relativas a um conjunto de aplicações para o âmbito do projeto que são atualizadas diariamente com nova informação através de *batchs* de extrações dos *snapshots* das fontes originais. As fontes de dados envolvidas estão estruturadas e caracterizadas de forma diferente, e a informação presente é muito dispersa, redundante e tem muito "lixo" associado.

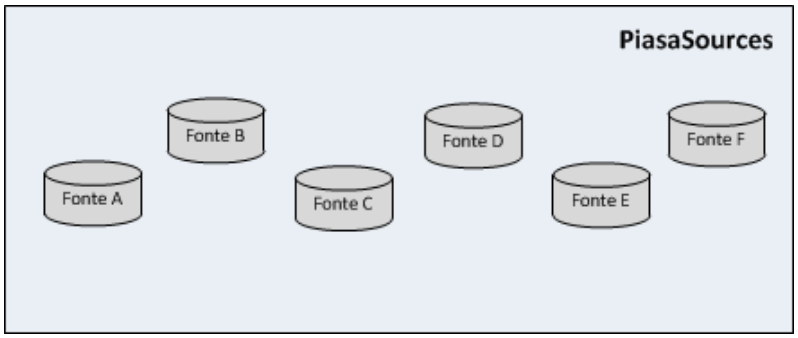

Figura 3.3 - Base de dados PiasaSources

No âmbito do projeto, considerou-se apenas evidências de acessos a aplicações de forma a reduzir a entrada de dados do projeto, mas futuramente o sistema incluirá outros tipos de evidências expandindo as fontes de dados envolvidas. Desta forma, foram consideradas contas aplicacionais de um conjunto de 10 aplicações de alto risco, perfazendo um total de 24 fontes de dados. Neste conjunto, encontram-se aplicações caracterizadas por muitos elementos de informação, assim como, por poucos, acerca do utilizador. Este número de fontes deve-se ao facto de serem consideradas além das próprias fontes das aplicações também as fontes das plataformas OIM e Pulso/Acessos para cada aplicação. Notar que algumas aplicações não são aprovisionadas pela plataforma OIM (e consequentemente pelo Pulso/Acessos), e desta forma, não haverá tal informação adicional para essas aplicações. Geralmente, cada aplicação dispõe de três fontes de dados, e portanto, serão utilizadas sempre que estas existirem:

- Fonte da própria aplicação – onde constam os registos sobre todas as contas de acesso existentes da respetiva aplicação, para *login* na aplicação.
- Fonte OIM da aplicação – onde consta a extração dos utilizadores com conta da aplicação que constam e foram aprovisionados pela plataforma OIM nessa aplicação nesta plataforma. Esta fonte permite caracterizar adicionalmente com maior detalhe a personalidade do utilizador dessa conta.
- Fonte Pulso/Acessos da aplicação – onde consta a extração dos pedidos de abertura de contas para uma qualquer aplicação na plataforma Pulso/Acessos. Tal como a fonte OIM, também esta permite adicionalmente caracterizar a personalidade do utilizador dessa conta, como por exemplo, quem criou e quando foi criada a conta.

Adicionalmente foi considerada também a própria fonte da plataforma de gestão de identidades da organização OIM que contém informação relevante sobre um conjunto de identidades já criadas até ao momento. A base de dados da plataforma OIM contém informação complementar à da AD.

3.2.2 Base de dados – InfoTemp

A base de dados InfoTemp é usada como uma passagem intermédia entre as bases de dados de entrada e de saída de dados, PiasaSources e PiasaInfo, respetivamente. Inicialmente foram criadas várias tabelas individuais deste tipo para cada fonte de dados, mas a longo prazo, com o aumento de fontes envolvidas tornou-se inexequível manter todas essas tabelas. Desta forma, foi criada uma única tabela gigante para armazenar todas as contas de todas as fontes de dados acabando por simplificar a gestão destas contas.

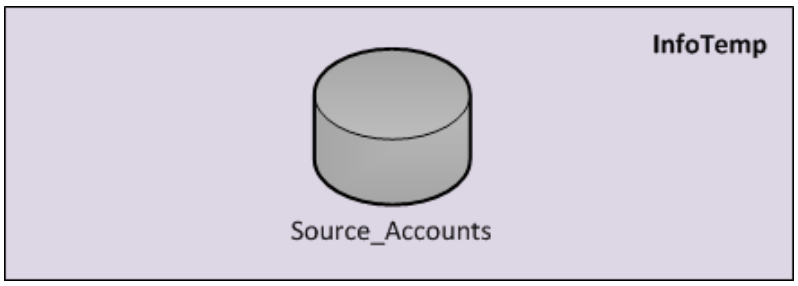

Figura 3.4 - Base de dados InfoTemp

Esta base de dados possui uma única tabela central canónica Source_Accounts que se destina a guardar temporariamente todas as contas de origem (registos de contas aplicacionais, registos da plataforma OIM e do Pulso/Acessos de cada aplicação, e identidades da plataforma OIM) provenientes das fontes de dados (base de dados PiasaSources) mas devidamente processadas no seu formato canónico resultantes do processo de ETL (descrito no subcapítulo 4.2), uma vez que estas estão estruturadas de forma diferente e são constituídas por diferentes elementos de informação. Os dados guardados nesta tabela são disponibilizados e utilizados no processo de sintetização de identidades (descrito no subcapítulo 4.3).

A tabela Source_Accounts é assim constituída:

Atributo	Descrição
Id	Identificador interno da tabela
Application_Id	Nome da aplicação da conta
Account_Id	*User_Id ou User_Login de acesso da conta*
Source_Acc_Table	Nome da fonte de dados (tabela de origem) da conta
Account_Type	Tipo de conta Os valores possíveis são: NormalUser, SysAdmin, DBA, Application, SysAdminTool, DBATool, MonitorTool, Unknown
Profile_Names	Nome dos perfis associados à conta
Basic_Risk_Level	Nível de risco da conta Os valores possíveis são: Normal, Reserved
Account_State	Estado da conta Os valores possíveis são: Active, Suspended, Removed
GUI_Match	Número de identificação do *ticket* do Pulso/Acessos

Creation_Date	Data de criação da conta
Last_Access_Date	Data do último acesso efetuado
FullName	Nome completo do utilizador da conta
FirstName	Primeiro nome
LastName	Último nome
MidName	Nome intermédio
Id_NM	Número mecanográfico ou colaborador PT
Id_NC	Número de identificação fiscal ou número de contribuinte
Id_BI	Número de bilhete de identidade ou cartão do cidadão
Id_PA	Número de identificação de passaporte
Id_AR	Número de autorização de residência
AD_User	Nome de utilizador de rede
AD_Domain	Domínio de rede da conta
AD_Email	Endereço de *email* do utilizador
Company_PT	Nome da empresa interna do utilizador
Company_Ext	Nome da empresa externa do utilizador
Association_PT	Nome da direção interna do utilizador
Sponsor_PT	Responsável pela abertura de conta
Source_Etl	Nome da rotina de ETL que deu origem à criação do registo

Tabela 3.1 - Tabela Source_Accounts

Todos os atributos listados são do tipo *typeless*, ou seja, podem armazenar quaisquer tipos de dados, para poderem lidar com possíveis erros de construção de valores proveniente das várias tabelas fontes, e apenas o atributo Profile_Names pode conter mais do que um valor.

3.2.3 Base de dados – PiasaInfo

A base de dados PiasaInfo é usada para guardar os resultados produzidos e sintetizados durante a execução do projeto. Dispõe de um conjunto de tabelas para armazenar toda a informação processada.

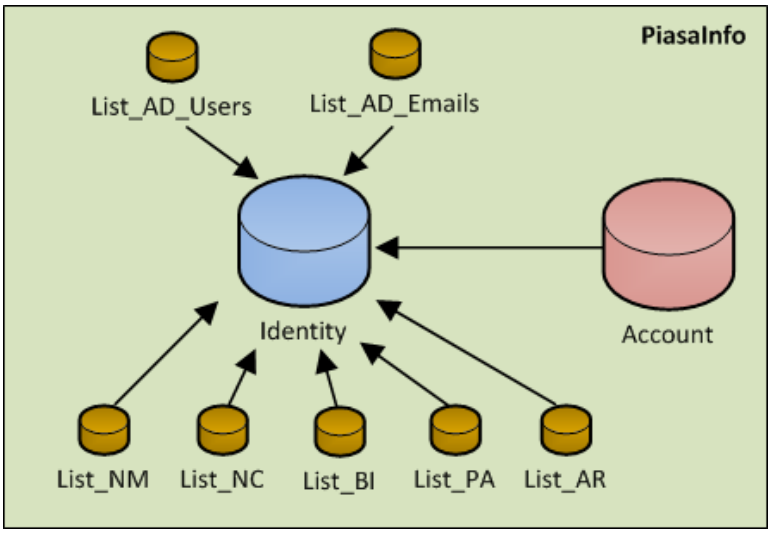

Figura 3.5 - Base de dados PisaInfo

Tal como a base de dados InfoTemp, também nesta existem atributos do tipo *typeless* para poderem lidar com possíveis erros de construção de valores, proveniente das diferentes tabelas fontes. As tabelas que incorporam esta base de dados são:

a) Identity

A tabela Identity é uma tabela central canónica usada para guardar toda a informação acerca das identidades sintetizadas, ou seja, toda a informação que constitui uma identidade ou personalidade mestre.

A tabela é assim constituída:

Atributo	Descrição
Identity_Id	Identificador interno da tabela
Identity_Type	Tipo de Identidade Os valores possíveis são: NormalUser, SysAdmin, DBA, Application, SysAdminTool, DBATool, MonitorTool, Unknown
FullName	Nome completo da Identidade
FirstName	Primeiro nome
LastName	Último nome
MidName	Nome intermédio
Id_NM	Número mecanográfico ou colaborador PT
Id_NC	Número de identificação fiscal ou número de contribuinte
Id_BI	Número de bilhete de identidade ou cartão do cidadão
Id_PA	Número de identificação de passaporte
Id_AR	Número de autorização de residência
Id_NC_Hash	Hash de NC
Id_BI_Hash	Hash de BI
Id_PA_Hash	Hash de PA
Id_AR_Hash	Hash de AR
AD_Users	Nome de utilizadores de rede
AD_Domains	Domínios de rede

AD_Emails	Endereços de *email* da identidade
Ad_Users_Num	Número total de utilizadores de rede
Ad_Emails_Num	Número total de endereços de *email*
Company_PT	Nome da empresa interna da Identidade
Company_Ext	Nome da empresa externa da Identidade
Association_PT	Nome da direção interna da Identidade
Sponsor_PT	Responsável pelas aberturas de contas
Creation_Date	Data de criação da Identidade
Last_Access_Date	Data do último acesso realizado por uma conta da Identidade
Last_Validation_Date	Data da última validação ou modificação da Identidade
Activity_Level	Número de acessos efetuados por dia
Accum_Risk_Level	Valor que relaciona o nº de contas com perfil de risco, nº de incidentes nos últimos 90 dias e nº *passwords* falhadas
Recomended_Action	Ação recomendada a realizar à Identidade Os valores possíveis são: N=None, V=Validate, M=Monitor, S=Suspend R=Remove, I=Inform
Matches_Num	Número de combinações da identidade
Tainted	Indicador de identidade "estragada" ou "contaminada". Significa que foi violada uma regra de unificação e não conseguiu realizar a combinação com outra identidade, requerendo assim intervenção humana. Os valores possíveis são: True, False
Annotations	Anotações informativas acerca da identidade
Source_Etl	Nome da rotina de ETL que deu origem à criação da identidade
Fail_Ids	Conjunto de identificadores de identidades que falharam a unificação com a identidade corrente
Tainted_Number	Número de identidades marcadas como *tainted* dos *clusters*

Tabela 3.2 – Tabela Identity

Como esta tabela vai guardar todos os detalhes que caracterizam uma identidade, alguns dos atributos aqui listados vão conter mais do que um único valor.

b) Account

A tabela Account é usada para guardar apenas todas as contas de acesso com diferentes perfis, sejam elas aplicacionais ou de bases de dados ou de sistema, associadas a uma identidade da tabela Identity, excluindo assim quaisquer registos provenientes da plataforma OIM e Pulso/Acessos. Isto porque um determinado evento de *login* referirá uma dessas contas. É uma tabela central canónica que armazena informação mínima que constitui uma conta ou personalidade operacional.

A tabela é assim constituída:

Atributo	Descrição
Id	Identificador interno da tabela
Application_Id	Nome da aplicação da conta
Account_Id	*User_Id ou User_Login* de acesso da conta

Source_Acc_Table	Nome da fonte de dados (tabela de origem) da conta
Account_Type	Tipo de conta Os valores possíveis são: NormalUser, SysAdmin, DBA, Application, SysAdminTool, DBATool, MonitorTool, Unknown
Profile_Names	Nomes de perfis associados à conta
Basic_Risk_Level	Nível de risco da conta
Identity_Id	Identificador da identidade associada à conta (referência ao atributo Identity_Id da tabela Identity)
Confidence_Level	Grau de certeza ou confiança da identidade associada à conta Os valores possíveis são: Strong, Weak
Confidence_Source	Origem da certeza ou confiança da identidade associada à conta Os valores possíveis são: PIASA_ILA, PIASA_ARM, Manual
Account_State	Estado da conta Os valores possíveis são: Active, Suspended, Removed
GUI_Match	Número de identificação do *ticket* da plataforma Pulso/Acessos
Creation_Date	Data de criação da conta
Last_Access_Date	Data do último acesso efetuado
Activity_Level	Número de acessos efetuados por dia
Accum_Risk_Level	Valor que relaciona o nº de incidentes nos últimos 90 dias
Recomended_Action	Ação recomendada a realizar à conta Os valores possíveis são: N=None, V=Validate, M=Monitor, S=Suspend, R=Remove, I=Inform
Annotations	Anotações informativas acerca da conta

Tabela 3.3- Tabela Account

Apenas o atributo Profile_Names pode conter mais do que um valor.

O atributo Confidence_Level representa o grau de credibilidade do relacionamento da identidade com a respetiva conta expresso através de dois valores, "Strong" e "Weak". Este nível indica a relação da personalidade operacional com a personalidade mestre associada. Antes do processo de unificação de registos, este atributo tem o valor "Strong" porque cada conta de origem é diretamente transformada numa personalidade mestre (identidade) e numa personalidade operacional (conta), existindo assim um relacionamento direto. Contudo, após o processo de unificação desta identidade com outra identidade, o atributo Confidence_Level é recalculado, isto é, tem o valor "Strong" se a unificação foi desencadeada por um dos cinco números identificativos (Id_NM, Id_NC, Id_BI, Id_PA ou Id_AR), caso contrário terá o valor "Weak" se forem usados os restantes atributos, sendo estes considerados menos credíveis para efeitos de unificação.

c) List_[NM | NC | BI | PA | AR | AD_User | AD_Email]

Cada uma destas tabelas guarda os respetivos tipos de valores utilizados por identidades, e ainda um conjunto de informação que os caracteriza. São tabelas centrais canónicas. Estas tabelas são assim constituídas:

Atributo	Descrição
Value	Valor do atributo
Recs	Conjunto de identificadores das identidades que usam o valor (referência ao atributo Identity_Id da tabela Identity)
Recs_Number	Número de identidades que usam o valor do atributo
Tainted	Indicador de valor "estragado" ou "contaminado". Significa que a unificação de identidades desencadeada por este valor de atributo falhou devido a uma violação das regras de unificação, requerendo assim intervenção humana. Os valores possíveis são: True, False
Error_Tag	Código de erro resultante da aplicação do módulo de validação de dados

Tabela 3.4 – Tabelas List

Apenas o atributo Recs pode conter mais do que um valor.

Todos os atributos indicados neste subcapítulo que agregam mais do um valor utilizam o carater vírgula para a separação de valores. Alguns desses atributos são inicialmente deste tipo mas grande parte aparecem somente após o processo de sintetização de identidades, mais especificamente durante a unificação ou combinação de atributos. No entanto, existem casos especiais de alguns atributos nunca armazenarem mais do que um valor, como por exemplo, Id_NM, Id_NC, Id_BI, Id_PA e Id_AR.

3.3 Arquitetura do sistema

Após delinear um modelo de dados capaz de representar e consolidar a informação que se pretende processar, passou-se para o desenho da arquitetura do sistema de modo a atingir os objetivos propostos. De seguida, podemos visualizar sob duas perspetivas a arquitetura do sistema.

3.3.1 Arquitetura de alto nível

Nesta perspetiva, analisando a figura 3.6 podemos visualizar a composição do *datawarehouse* abstraindo os componentes intermediários de informação. Analisando a seguinte figura, podemos visualizar claramente a contribuição de cada fonte de dados (base de dados PiasaSources) para a criação de uma identidade e de uma conta (base de dados PiasaInfo).

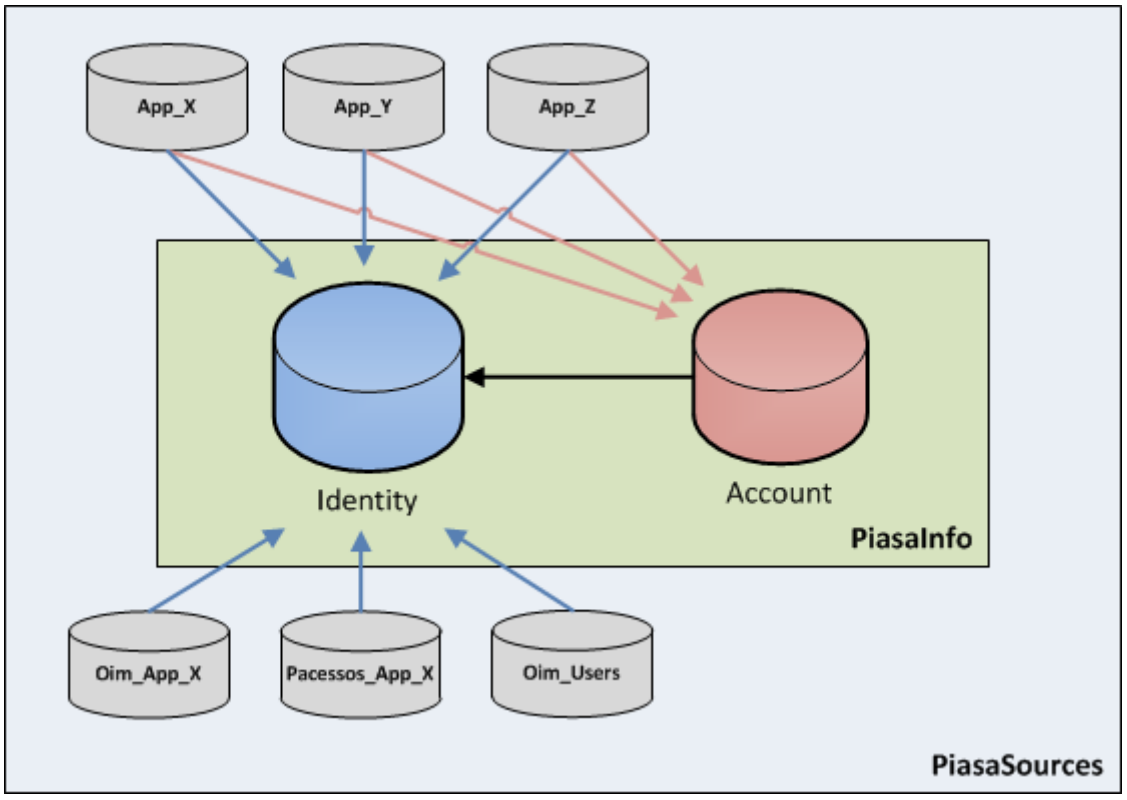

Figura 3.6 - Arquitetura do sistema de alto nível

3.3.2 Arquitetura detalhada

Nesta perspetiva, analisando a figura 3.7 podemos ver todas as bases de dados descritas anteriormente envolvidas, o fluxo de informação entre ambas, e ainda os processos de operações realizados, que são os seguintes:

- Processo de ETL, responsável pela extração dos registos provenientes das fontes de dados no formato canónico, transformação para o seu formato canónico e carregamento para a tabela central canónica Source_Accounts. Processo detalhado no subcapítulo 4.3.
- Processo de Sintetização, responsável pela unificação de identidades e atributos e aglomeração das respetivas contas. Processo detalhado no subcapítulo 4.4.
- Processo de Validação, responsável por validar a estrutura e estado de validade dos valores utilizados nas identidades sintetizadas. Processo detalhado no subcapítulo 4.5.

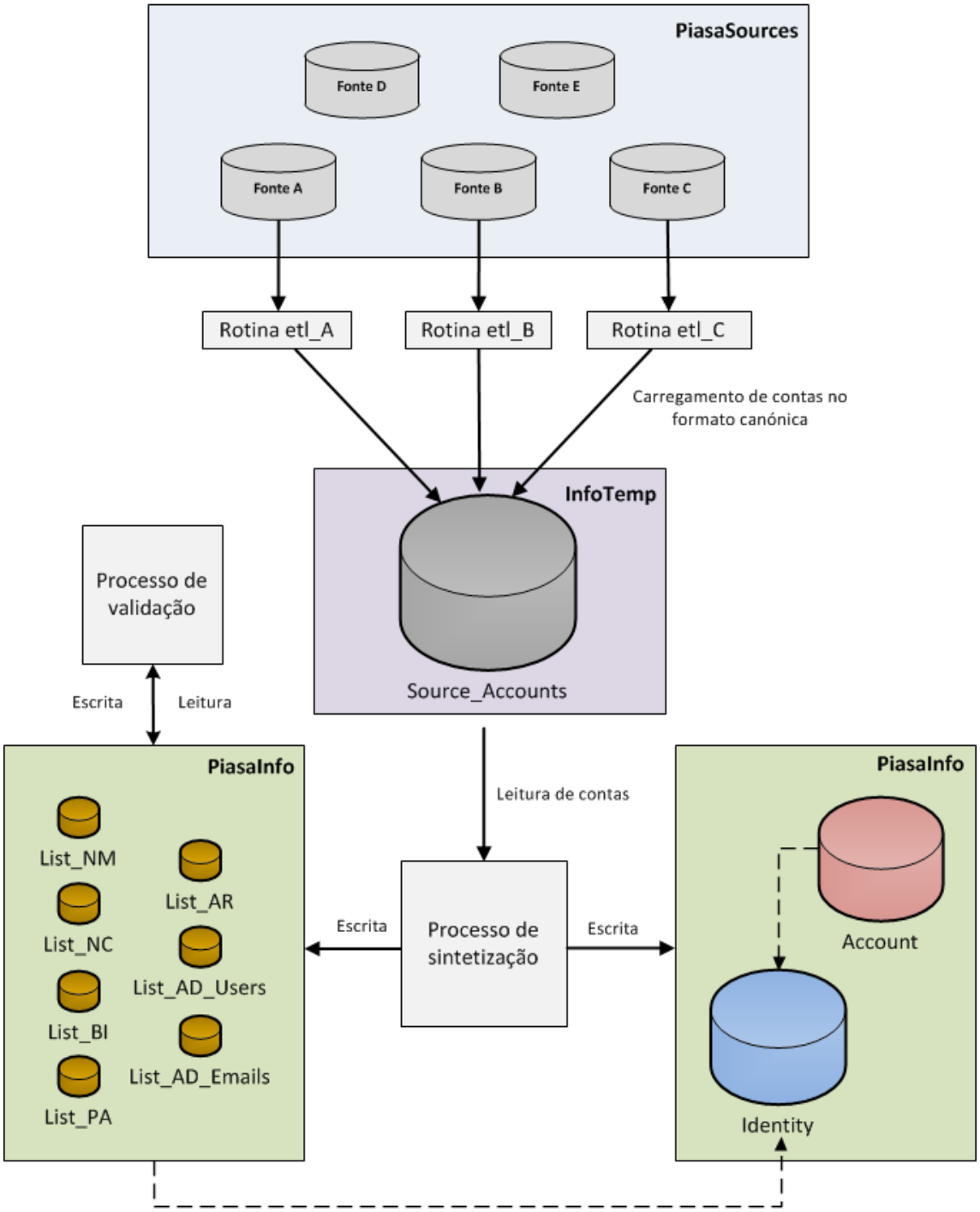

Figura 3.7 - Arquitetura do sistema detalhada

Capítulo 4

Concretização

Neste capítulo encontra-se documentado o desenvolvimento da solução proposta para alcançar os objetivos especificados no capítulo 1. São apresentadas as especificações das funções de similaridade a usar na sintetização de identidades, de seguida, a modelação do sistema através de um diagrama de classes UML, a caracterização detalhada dos complexos processos envolvidos na solução, nomeadamente, processo de ETL, processo de sintetização de identidades e processo de validação de dados. Por último, é apresentada a plataforma desenvolvida em linguagem Ruby que engloba todos estes componentes, designado por PiasaServer.

4.1 Funções de similaridade

De modo a construir um diagrama de classes UML e resolver entidades durante o processo de sintetização de identidades foi importante previamente especificar funções para medir o grau de semelhança entre quaisquer dois valores de atributos, quando necessário.

Entre o leque de métricas de similaridade investigadas no subcapítulo 2.2.3 foi escolhida a métrica de distância baseada em carateres Jaro Winkler Distance. Esta métrica foi sugerida por vários investigadores que demonstraram ter um bom desempenho quando se pretende avaliar a similaridade entre duas *strings* (sequências de carateres). Como descrito nesse mesmo capítulo, o algoritmo desta métrica começa por determinar a quantidade de elementos em comum (caracteres na mesma posição) nas *strings* e utiliza a quantidade como base para uma fórmula própria para calcular a similaridade. Inclui ainda uma correção do valor final da comparação de forma a evitar que *strings* que sejam diferentes apenas na parte final tenham uma grande distância entre ambas. O resultado da aplicação deste algoritmo consiste num valor decimal entre 0 e 1, representando assim a similaridade entre as *strings*. O valor 1 significa que as *strings* são idênticas e 0 significa que são completamente diferentes. O valor para o seu

threshold (limite) foi 0.75, resultando de inúmeras experiências que permitiram a sua escolha.

Os atributos que caracterizam uma identidade alvo de comparações por similaridade são os seguintes:

a) Nomes completos (atributo FullName)

Antes de qualquer comparação é efetuada a colocação dos dois nomes num formato comum a ambas, designado por canónico, para que as duas *strings* estejam em igualdade. A transformação canónica corresponde a um conjunto de operações de manipulação do seu conteúdo, que são as seguintes:

1) Conversão para letras minúsculas;
2) Remoção de acentos ou cedilhas, ou seja, substituição pelos mesmos carateres mas sem acento ou cedilha;
3) Remoção de:
   - Carateres do conjunto {-.,;_};
   - Separadores da forma "\n";
   - Espaços em branco;
   - Palavras dentro de parêntesis, pelicas ou aspas;
   - Palavras compostas por um único carater apenas, por exemplo, "C." ou "C";
   - Preposições portuguesas do conjunto {dos, das, do, da, de, e}.

Após a transformação para o formato canónico das duas *strings* procede-se então à aplicação da métrica de similaridade escolhida.

b) Endereços de *email* (atributo AD_Email)

Pelas mesmas razões acima também é necessário escolher um formato comum aos dois endereços. Neste tipo de atributo apenas interessa comparar a parte relativa ao nome do endereço, excluindo assim o domínio do mesmo. As operações de transformação para formato canónico são as seguintes:

1) Conversão para letras minúsculas;
2) Substituição de carateres "." e "-" por espaços;
3) Remoção de acentos ou cedilhas, ou seja, substituição pelos mesmos carateres mas sem acento ou cedilha;
4) Remoção de palavras compostas por um único carater apenas, por exemplo, "c".

De seguida, procede-se então à aplicação da métrica de similaridade escolhida.

4.2 Diagrama de classes UML

O desenho de um diagrama UML (Unified Modeling Language) foi extremamente útil na fase inicial de desenho permitindo modelar o vocabulário do sistema do ponto de vista da solução. O diagrama foi refinado ao longo de várias etapas do desenvolvimento e demonstra as diferentes classes que compõem o sistema assim como se relacionam entre si. O desenho foi baseado no modelo de dados descrito no subcapítulo 3.2, e segundo o paradigma orientado a objetos as classes criadas são:

- A classe **SourceAccount** representa um registo da tabela temporária canónica Source_Accounts. Os principais métodos desta classe são:
  - to_identity – permite converter para um objeto do tipo Identity com os respetivos atributos.
  - to_account – permite converter para um objeto do tipo Account com os respetivos atributos.
- A classe **Identity** representa um registo da tabela Identity. Além dos atributos dessa tabela, existe um atributo adicional "accounts" que permite armazenar todas as contas que uma identidade possui, ou seja, uma lista de objetos do tipo Account.
- A classe **Account** representa um registo da tabela Account.
- A classe **Name** representa um nome completo de uma identidade ou utilizador identificando o primeiro, intermédio e último nome, e ainda permite comparar ou medir a similaridade de dois nomes recorrendo a técnicas de similaridade. Os métodos desta classe são:
  - clean_name – coloca um nome no formato canónico descrito subcapítulo 4.1, aplicando um conjunto de operações de limpeza e tratamento.
  - is_similar – determina se dois nomes são similares, após a colocação dos nomes no formato canónico usando a função anterior. É uma função hibrida, isto é, verifica inicialmente por comparação de inclusão determinando se o nome menor está ou não incluído no nome maior (baseada em *tokens*), e caso esta estratégia não seja suficiente recorre-se à métrica de distância baseada em caracteres, Jaro-Winkler Distance, descrita no subcapítulo 4.1.
- A superclasse **UserID** representa um valor de identificação usado por uma identidade. Esta classe é usada intensivamente nos índices invertidos no processo de deteção e combinação de registos duplicados no processo do subcapítulo 4.3. As subclasses NM, NC, BI, PA, AR, ADUser e ADEmail representam

respetivamente um registo das tabelas List_NM, List_NC, List_BI, List_PA, List_AR, List_AD_User e List_AD_Email. O principal método desta classe é:

- add_record – adiciona uma identidade ao objeto UserID significando que essa identidade utiliza o valor do atributo "value" desse objeto UserID. No fundo, é adicionado à lista de identidades um objeto do tipo Identity ao objeto UserID corrente.

Ainda, existe a subclasse FullName mas esta não tem qualquer referência a uma tabela, sendo apenas usada para deteção e combinação de duplicados. A subclasse ADEmail representa um valor de endereço de *email* e possibilita a comparação de dois endereços. Os principais métodos desta subclasse são:

- clean_email_name – coloca o endereço no formato canónico descrito no subcapítulo 4.1, aplicando um conjunto de operações de limpeza e tratamento ao nome do endereço.
- is_similar – determina se dois endereços são similares, focado apenas na parte do nome do endereço (excluindo o domínio), após a colocação do endereço no formato canónico usando a função anterior. A operação consiste na verificação por comparação de inclusão determinando se o nome menor está incluindo no nome maior (baseada em *tokens*).
- compare_email_with_name – determina se um nome de uma identidade é similar ao nome usado do endereço de *email* corrente. Após colocação dos nomes no formato canónico, determina se o nome usado no endereço está incluído no nome completo.

- A classe **MySQLConnector** representa uma ligação ao servidor de MySQL permitindo um conjunto de operações sobre as tabelas relacionais, agregando todas as operações necessárias para leitura e escrito em bases de dados de forma bastante eficiente recorrendo à utilização de *batchs* de *statements* e otimizando assim as ligações às bases de dados. As operações de inserção na base de dados estão equipadas para lidar com a prevenção de injeção de SQL. É utilizado o *driver* JDBC[25] para MySQL para a conexão das bases de dados.

A seguinte figura representa todas as relações entre as classes criadas.

[25] http://www.mysql.com/products/connector/

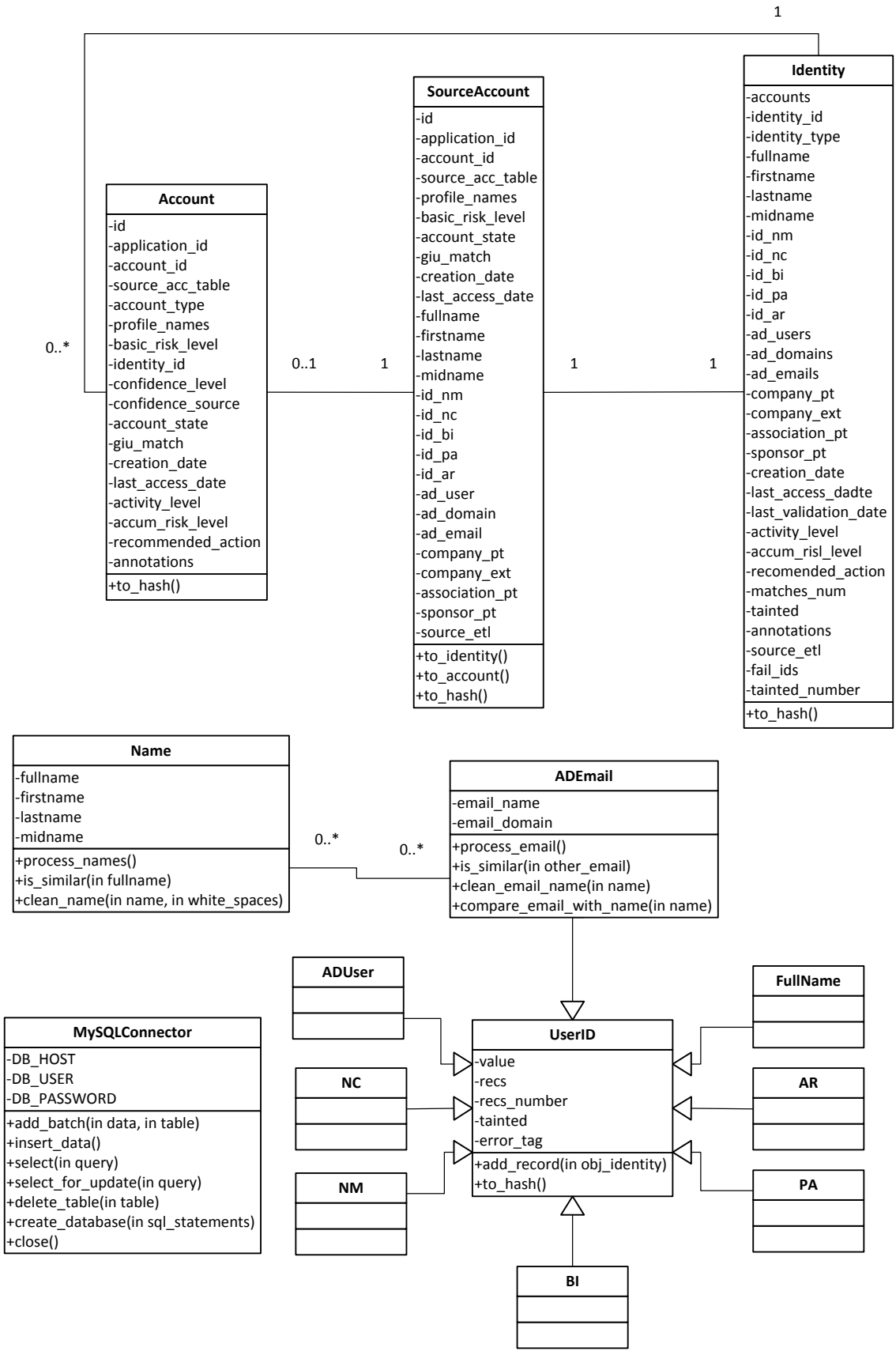

Figura 4.1 - Diagrama de classes UML

4.3 Processo de ETL

Como vimos no subcapítulo 2.2.3, é muito importante realizar uma fase de preparação e limpeza de dados antes de iniciar o processo de deteção de registos duplicados quando se pretende resolver entidades. Desta forma, é utilizado um processo de Extração, Transformação e Carregamento ou simplesmente ETL (Extract, Transform and Load) sendo responsável por essa fase. Neste capítulo, são apresentadas as vantagens e benefícios da utilização de um processo de ETL no contexto do projeto, a análise realizada ao sistema das fontes de dados envolvidas passando por uma fase de descoberta de dados e deteção de anomalias, e por último, a descrição detalhada da solução desenvolvida para este processo.

4.3.1 Vantagens de um processo de ETL

No geral, este tipo de procedimento é muito utilizado em *datawarehouses* que consolidam dados de diferentes fontes onde atualizações frequentes de vários sistemas são agregadas e refinadas para uma posterior análise usando ferramentas especializadas. Estes processos tipicamente são componentes reusáveis que podem ser programados para serem executados regularmente ou quando novos dados aparecem na fonte de dados, e na maioria dos casos, essas fontes tendem a ser bases de dados relacionais ou ficheiros de texto. É também bastante utilizado em outras áreas como Data Quality, Business Intelligence, Data Migration, Application Integration, entre outros.

Segundo os autores de [24], o ETL deve ser um componente integral, contínuo e recorrente num *datawarehouse*, e ao mesmo tempo, um processo automatizável, bem documentado e facilmente mutável.

No contexto do projeto, o sucesso do *datawarehouse* requer que o processo de ETL esteja fortemente ligado ao processo de qualidade de dados com o objetivo de produzir informação correta, coesa, consistente e completa, e é esperado processar toda a informação recebida, normalizando-a em formatos uniformes, categorizando-a em colunas corretas e assim possibilitar a criação de um novo formato dados, designado por formato canónico, melhorando assim o nível de qualidade da informação. Quando se pretende atingir uma boa qualidade de dados, o desempenho de um processo de ETL é fortemente reduzido.

Por omissão, um processo de ETL está associado a três operações distintas:

- Extração de dados proveniente de uma ou mais *data sources* (fontes de dados).
- Transformação de dados, que em geral, consiste em funções de limpeza, reformatação, padronização, agregação e aplicações de regras de negócio específicas. Esta fase é considerada a mais crítica porque pode ser bastante complexa levando a problemas operacionais significativos.

- Carregamento de novos dados para outro repositório (sistemas de destino específicos ou ficheiros).

Como se pode ver, o ETL está ligado a uma grande combinação de processos e tecnologias que podem ter um custo computacional significativo no desenvolvimento de um *datawarehouse*, e por isso, deve ser realizado de forma otimizada. Para diminuir o consumo computacional, as operações de ETL devem ser executadas sobre uma base de dados relacional separada da base de dados de origem dos dados (PiasaSources) e da base de dados final do *datawarehouse* (PiasaInfo), sendo utilizada portanto a base de dados InfoTemp com a tabela "Source_Accounts", para este efeito.

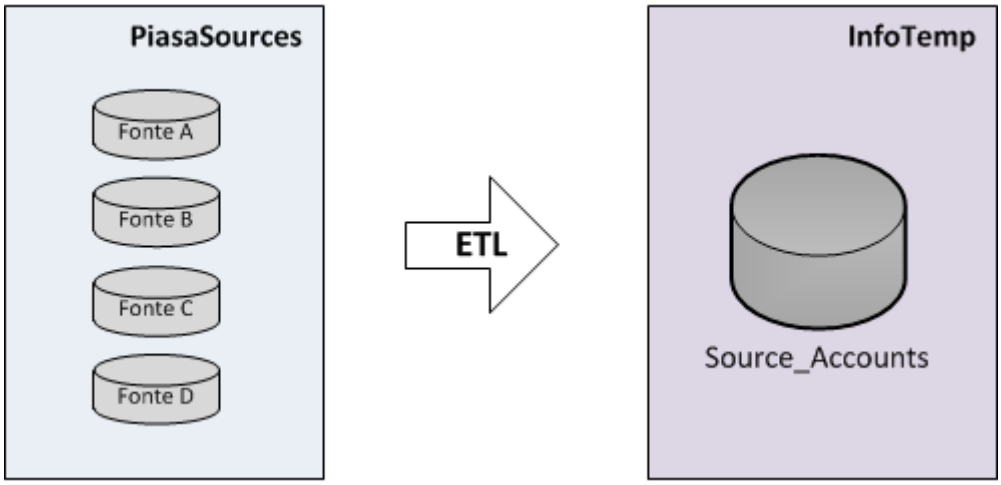

Figura 4.2 - Processo ETL no contexto do projeto

A utilização deste processo permite assim resolver o problema de heterogeneidade estrutural existente nas várias fontes provenientes dos diferentes sistemas, obrigando previamente a efetuar uma análise a todas as fontes de dados, a qual está descrita de seguida.

4.3.2 Análise das fontes de dados

Antes do desenvolvimento do processo de ETL propriamente dito foi essencial e necessário realizar um estudo das diversas fontes a utilizar como entrada de dados do projeto, como concluído anteriormente. Neste estudo, para cada fonte foi analisado de forma cuidada a contribuição de cada coluna (atributo) e os possíveis valores armazenados, de forma a compreender e escolher os atributos considerados relevantes para o modelo de dados definido no subcapítulo 3.2, e para o correto mapeamento de atributos.

Durante a análise foram detetadas anomalias e identificados erros de construção de valores e algumas observações importantes a ter em consideração na fase de desenvolvimento do processo, nomeadamente na etapa de transformação. Para uma

correta categorização e mapeamento dos valores dos atributos foram utilizadas regras específicas para cada fonte com base nesta análise. Algumas das observações encontradas foram:

a) Existência de contas que possuem um endereço de *email* diferente do nome do utilizador da respetiva conta. Isto pode acontecer devido ao facto de que no momento do registo da conta foi utilizado o endereço da pessoa que autorizou a abertura da conta, ou simplesmente um erro de introdução dos dados. Por exemplo: uma conta da aplicação "AppX" com nome de utilizador "João Miguel Pedro" com o endereço de *email* sara-marques@telecom.pt.

b) Existência de atributos com diferentes tipos de valores misturados, como por exemplo, atributo de *login* ou *user_id* que refere o acesso a uma aplicação pode utilizar valores alfanuméricos com variados significados. Alguns desses valores são, por exemplo:

- Valores que referem números de documentos identificativos com prefixo do tipo de documento:
  - "BI12345678"
  - "NM12345678"
  - "PT12345678"
  - "_BI1234567_OLD"
  - "_OLDBI12345678_OLD"
- Valores que referem nomes de utilizadores de rede com diversos prefixos:
  - "EXT-ABCDE"
  - "DOMAIN\ ABCDE"
- Valores aleatórios sem qualquer informação adicional acerca do seu conteúdo:
  - "JMiguel"
  - "1111234"
  - "_0000001111"
  - "xxx123456789xxx"
  - "18238"

Os atributos que guardam informação de acessos ou *logins* (atributos de nome "User_Id" ou "User_Login") de aplicações são úteis para a construção das identidades pois são capazes de fornecer informação extra ou complementar, que pode não existir em mais nenhum outro atributo. Por exemplo, através de uma conta, poder-se-á obter o valor do atributo Id_BI filtrando o atributo de *login*. Os atributos que representam um tipo de valor específico, como por exemplo, atributo "Número_Mecanográfico", por vezes, contêm além desses mesmos valores outros valores como Bilhete de Identidade

ou Número Contribuinte. Esta anomalia pode resultar na incorreta introdução de valores por parte do ser humano. Para detetar tais anomalias desenvolveu-se um módulo de funções de Validação de dados, que se encontra descrito no subcapítulo 4.5.

Após a análise das fontes e dos casos anómalos encontrados concluiu-se que uma vez que as fontes estão estruturadas de forma diferente entre si (usando diferentes colunas para representar a mesma informação) e utilizam diferentes formatos para representar um mesmo valor, decidiu-se desenvolver uma rotina de ETL individual e independente para cada fonte de informação, não sendo possível assim a implementação de uma rotina genérica para todas as fontes. E portanto, o processo de ETL resulta na execução de um grupo de rotinas específicas.

Fontes de dados

No âmbito do projeto foram selecionadas 10 aplicações com diferentes níveis de informação, incluindo a pseudo aplicação AD. Neste conjunto encontram-se fontes constituídas por muitos elementos informativos assim como fontes com muito pouca informação acerca do seu utilizador, caracterizando assim a sua personalidade operacional. A pseudo aplicação AD contém muita informação acerca da personalidade da conta de rede, ao contrário de outras aplicações com pouca informação, sendo assim difícil identificar a identidade a que pertencem essas contas. Neste tipo de contas com pouca informação, existem casos em que apenas poderemos utilizar o nome como elemento de comparação e pouco mais, e em algumas situações o nome apenas é constituído pelo primeiro e último nome. Esta mistura de diferentes níveis de informação permitiu medir o grau de desempenho e precisão do algoritmo desenvolvido no processo de sintetização de identidades, de forma a verificar a precisão de identificação da personalidade mestre (identidade) de uma conta caracterizada por pouca informação.

Desta forma, o processo de ETL conta com cerca de 24 rotinas específicas para cada aplicação, incluindo as fontes das próprias aplicações, e ainda das plataformas Pulso/Acessos e OIM para cada aplicação, se existir. Adicionalmente foi também considerada a fonte da própria plataforma OIM que diz respeito a registos de identidades, sendo extremamente importante no processo de sintetização de identidades para detetar erros e eliminar redundâncias.

A título de exemplo, podemos ver como são aplicadas as rotinas de ETL à pseudo aplicação AD, existindo no total três rotinas associadas.

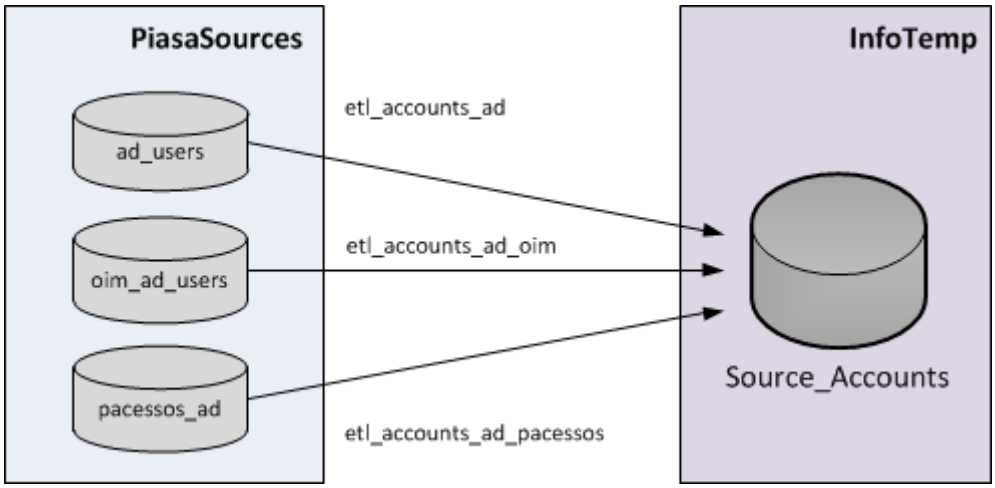

Figura 4.3 - Exemplo de processo ETLL aplicado à pseudo-aplicação AD

4.3.3 Descrição do processo de ETL

O processo de ETL a desenvolver é exclusivamente focado em contas aplicacionais com acesso a informação reservada do conjunto de aplicações indicado anteriormente, e, resultará num formato canónico e numa melhor qualidade dos dados. Durante a análise realizada constatou-se a existência de heterogeneidade estrutural (diferentes colunas para representar a mesma informação) e lexical (diferentes formas de representar o mesmo valor) presente nas fontes de informação, e, desta forma, foram criadas rotinas de ETL individuais para implementação de regras específicas, tanto para extração de valores para obtenção de informação como como para mapeamento de atributos para cada fonte de dados.

A constituição típica de um processo de ETL engloba as seguintes etapas e respetivas operações [24]:

a) Extração

Etapa de extração dos registos provenientes das várias fontes de dados presentes na base de dados PiasaSources. Cada rotina de ETL recebe informação *raw* contendo erros e formatos não normalizados. As contas a processar dizem respeito apenas a contas ativas, excluindo assim contas obsoletas e inativas, isto porque os eventos de *login* que poderão surgir referem apenas contas que estejam ativas. Em algumas fontes não existe explicitamente um atributo que indique o estado da conta e então foi necessário implementar um mecanismo adicional, quando aplicável, para perceber se está ativa ou obsoleta, passando por uma comparação de datas, entre a data de validade da conta e a data atual em que nos encontramos. Por exemplo, os seguintes dois excertos SQL da cláusula WHERE seleciona apenas contas ativas sem conhecimento do estado das mesmas:

```
    SELECT … WHERE STR_TO_DATE(VALID_TO, '%Y-%m-%d %H:%i:%s') >= NOW()
    SELECT … WHERE DATE_FORMAT(STR_TO_DATE(ENDDA, '%Y%m%d'), '%Y-%m-%d') >=
CURDATE()
```

Durante a análise verificou-se ainda existência de contas duplicadas em algumas fontes de dados, o que levou à aplicação da cláusula SQL GROUP BY permitindo o agrupamento de registos a fim de remover eventuais duplicados (contas que referem uma mesma conta mas com diferentes perfis de acesso) e assim reduzir a quantidade de registos envolvidos ou processados. Nomeadamente foi feito o agrupamento nos seguintes tipos de fontes de dados dos seguintes atributos:

- Fontes de aplicações: agrupamento pelo atributo de *user_login* ou *user_id*;
- Fontes de Pulso/Acessos de cada aplicação: agrupamento pelo atributo de NomeDestinatario;
- Fontes de OIM's de cada aplicação: agrupamento pelo atributo *usrlogin*.

Adicionalmente, foi desenvolvida uma operação para algumas fontes de dados que durante a sua análise demonstraram a utilização de endereços de *emails* que não pertencem ao respetivo utilizador. Desta forma, procedeu-se à construção para cada fonte de uma *black list* (lista negra) de endereços de *email* para fazer tal distinção. Esta operação verifica se os endereços de *email* usados numa fonte dizem respeito ao próprio utilizador da conta ou ao responsável pela sua abertura, sendo apenas usados (ou obtidos) nas identidades a sintetizadas os endereços que não estejam na *black list*. O método escolhido para determinar se um endereço de *email* vai para a *black list* consiste em verificar quantas utilizações em diferentes contas tem cada endereço e caso seja superior a uma será considerado um candidato para a *black list*, e portanto não é considerado para a sintetização de identidades. Esta operação de *black list* contribui assim para uma melhor qualidade dos dados produzidos. A consulta SQL realizada antes da obtenção das contas ativas para construção destas listas para essas fontes de dados é a seguinte:

```
SELECT EMAIL FROM (
    SELECT count(*) AS TOTAL, EMAIL, FULLNAME FROM (
       SELECT EMAIL, FULLNAME
       FROM source_application_one
      WHERE EMAIL IS NOT NULL
      GROUP BY NAME
   ) AS mytable2 GROUP BY EMAIL
) AS mytable WHERE TOTAL > 1
```

Durante a análise das fontes de dados do Pulso/Acesso, a obtenção de endereços de *email* foi descartada devido a estes poderem dizer respeito aos próprios utilizadores como aos seus responsáveis pela abertura de conta, não existindo uma forma para fazer a distinção.

Para cada rotina foi elaborado um mapa lógico de dados antes de proceder à transformação dos dados físicos. Este mapa descreve a relação entre os atributos de origem e de destino, das bases de dados PiasaSources e InfoTemp, respetivamente, e foi representado através de uma tabela, como podemos ver de seguida.

Origem			Destino			Transformação
Tabela	**Coluna**	**Tipo_Dados**	**Tabela**	**Coluna**	**Tipo_Dados**	**Regra**

Figura 4.4 - Tabela de mapeamento lógico

O mapeamento lógico foi um componente importante e necessário para tornar eficientes os processos de ETL, tendo como objetivo transmitir o que se espera do processo, representando a sequência de operações envolvidas na etapa de transformação. E como tal, foi criado um mapa lógico de dados para cada fonte de dados baseado na análise realizada anteriormente.

b) Transformação

Segundo William E. Winkler [20], os métodos a usar na padronização ou uniformização de dados podem ser baseados em regras, pois, geralmente estas regras são desenvolvidas partindo de um conjunto de testes. Desta forma, com base no estudo feito anteriormente conseguiu-se elaborar um conjunto de regras de classificação e mapeamento dos atributos usados em cada fonte. Algumas destas regras são de mapeamento direto de atributos e outras mais complexas envolvendo a manipulação do próprio conteúdo antes do mapeamento propriamente dito.

Os valores de origem nunca são modificados, pois um dos objetivos deste projeto consiste em validar esses mesmos valores para posterior correção ou limpeza desses dados através de intervenção humana.

Regras de mapeamento direto

Na maior parte dos casos, as regras de transformação usadas são de mapeamento direto de atributos. Tal acontece, quando determinados atributos possuem exclusivamente valores específicos, não sendo necessário aplicar operações de manipulação. Por exemplo, se tivermos a coluna Name = “João Miguel Costa” então podemos inferir o atributo FullName = “João Miguel Costa”. Outro exemplo muitas vezes usado consiste em inferir o atributo FullName através da junção de atributos que

refiram partes do nome, como primeiro e último nome. A padronização do atributo FullName consiste então no processamento do primeiro e último nome, nome intermédio, e ainda em nome completo.

Id	FullName
1	João Miguel Costa Andrade
2	Luís Gomes
3	Maria Duarte Santos

Tabela 4.1 –Atributos FullName

Id	FullName	FirstName	LastName	MidName
1	João Miguel Costa Andrade	João	Andrade	Miguel Andrade
2	Luís Gomes	Luís	Gomes	
3	Maria Duarte Santos	Maria	Santos	Duarte

Tabela 4.2 - Padronização do atributo FullName

Regras de mapeamento indireto

As regras de mapeamento indireto envolvem a manipulação de valores e nem sempre o cenário é simples. Existem atributos na origem que requerem a manipulação do conteúdo dos seus valores antes do mapeamento no atributo de destino. As operações de manipulação possibilitam a limpeza de carateres desnecessários (como a remoção de carateres como "_", "OLD", "EXT_" e espaços em branco no início e no fim) e são baseadas em regras específicas documentadas nas respetivas fontes de dados, para a correta interpretação de valores. Por exemplo, atributos do tipo *User_Id* ou *User_Login* (que dizem respeito ao acesso usado numa aplicação) após serem filtrados fornecem informação extra ao *datawarehouse*. Se existe o atributo User_Login = "BI12345678" então podemos inferir o atributo Id_BI = "12345678", usando a regra de exclusão do prefixo "BI". O mesmo acontece para outro tipo de prefixos similares. Por exemplo, quando temos User_Login = "TMN\XPTO" podemos inferir não só o atributo AD_User = "XPTO" mas também o atributo AD_Domain = "TMN", usando a regra apropriada.

Para o mesmo tipo de atributo, existe outro cenário mais complexo que é quando o valor usado na fonte refere um número identificativo aleatório. Neste caso, são usadas regras documentadas das próprias fontes que permitem "decifrar" qual o tipo de documento a fim de perceber de que número se refere dentro do conjunto de números de identificação utilizados pela organização. Por exemplo, as seguintes regras permitem mapear corretamente o valor usado na origem:

- O número refere o atributo Id_NM se composto por 8 dígitos e iniciado por "100";

- O número refere o atributo Id_NC se composto por 9 dígitos;
- O número refere o atributo Id_PA se inclui pelo menos uma letra;
- O número refere o atributo Id_BI se composto entre 0 a 8 dígitos;
- O número refere o atributo AD_User em último caso.

Ainda existe um outro caso relacionado com o nome da empresa a que está afeto um utilizador. Algumas fontes não possuem informação suficiente para indicar se uma empresa é interna ou externa. Perante este cenário, é usada uma regra que aplica uma expressão regular a um nome de uma empresa, e desta forma, consegue-se inferir se a empresa refere o atributo Company_PT ou Company_Ext.

Estas são apenas as principais regras que foram elaboradas.

c) Carregamento

Etapa de carregamento dos novos dados devidamente tratados para a tabela canónica Source_Accounts, da base de dados InfoTemp.

De seguida, podemos ver um exemplo do mapeamento lógico de atributos para a aplicação "AppX", utilizando a respetiva rotina de ETL, "etl_accounts_appX".

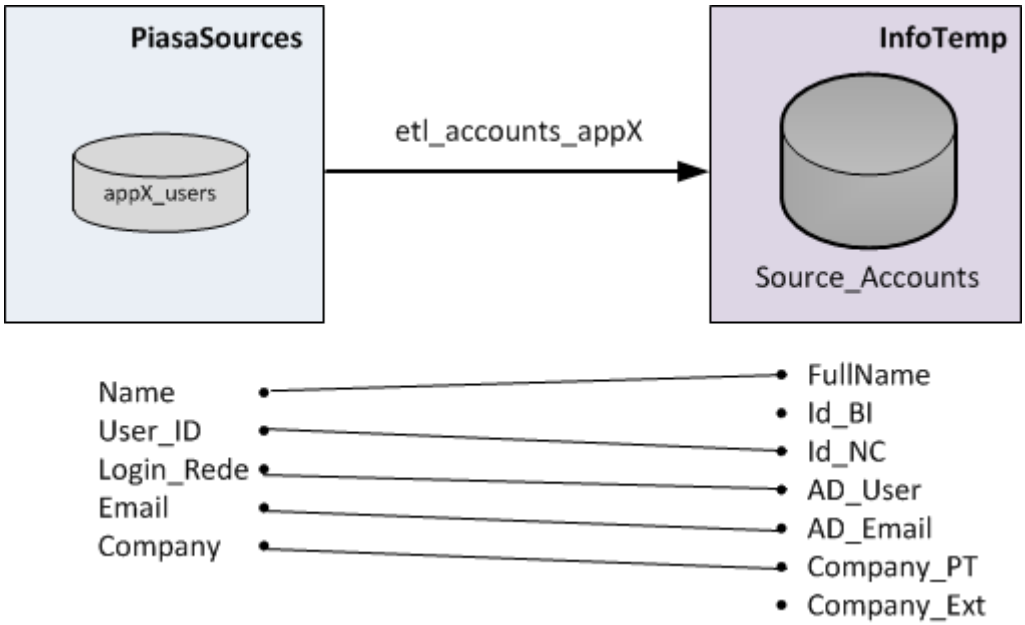

Figura 4.5 - Mapeamento lógico de atributos

Para concluir esta fase, o processo de ETL apenas trabalha com registos ou contas ativas, e assim as identidades sintetizadas pela solução são consideradas também identidades ativas. Foi criada uma biblioteca de funções "ETL_Functions" que auxiliam o processamento destas rotinas de ETL.

4.4 Processo de sintetização de identidades

O processo de sintetização ou unificação de objetos, que no contexto do projeto são identidades, é um procedimento bastante complexo e caro computacionalmente pois é nesta componente que é realizado o agrupamento ou unificação de identidades partindo das diferentes contas aplicacionais disponíveis. Consequentemente é manipulada uma enorme quantidade de registos (na ordem das centenas de milhares) em memória, e por isso, é necessário adotar delicadamente estratégias e técnicas para manter o processo eficaz e sobretudo eficiente.

A forma mais simplista (ou também conhecida por naïve) de combinar registos consiste em comparar cada registo de uma tabela A com cada registo de uma tabela B, mas infelizmente esta estratégia é extremamente custosa computacionalmente mesmo para tabelas de pequeno tamanho, requerendo um total de |A| . |B| comparações. A complexidade temporal para este algoritmo é quadrática, ou seja, $O(n^2)$. Como um dos objetivos consiste em desenvolver uma solução extremamente eficiente não foi implementada o algoritmo naïve devido a não ser viável em tempo útil, e em alternativa implementar estratégias que reduzem o número de comparações de registos e melhoram a eficiência da comparação de registos.

Este processo é então dividido em duas fases fundamentais:

a) Deteção de registos duplicados;

b) Combinação de registos duplicados.

4.4.1 Método para deteção de registos duplicados

O procedimento para detetar registos numa base de dados que são aproximadamente duplicados, mas não necessariamente idênticos, é um problema bastante comum em ambientes onde bases de dados com diferentes esquemas devem ser integradas, e onde os registos contêm erros ou falta de informação, que é o que se pretende resolver com este projeto. Como vimos anteriormente no subcapítulo 2.2.3, as técnicas de combinação de pares de registos requerem um algoritmo para deteção de relações do tipo “A é um duplicado de B” para um par de registos A e B, e devem ser específicos de aplicações, usando regras baseadas em domínios de conhecimento específico. A eficiência destes algoritmos depende muitas vezes das técnicas e estruturas de dados utilizadas.

Foram então analisadas várias soluções para o problema, implementando técnicas e estruturas de dados que envolvem diferentes abordagens, chegando a uma solução que permitiu reduzir eficazmente o tempo de execução e a quantidade de memória usada. Geralmente, a forma mais comum para identificar duplicados numa tabela é ordenar a

tabela segundo um determinado atributo ou chave para que os registos apareçam próximos uns dos outros. E de seguida, verificar se pares de registos vizinhos ou próximos são duplicados fazendo comparações emparelhadas de registos próximos, deslizando uma janela de tamanho fixo. Este tipo de estratégia demonstrou ser muito pesado computacionalmente, e por isso, em alternativa recorreu-se a técnicas de agrupamento a fim de reduzir o número de vezes que o algoritmo de comparação deve ser aplicado.

Como alternativa, foi utilizado um método para fazer um agrupamento preliminar de registos, em que cada registo é considerado separadamente como um *source record* (registo de origem) e usado para consultar os restantes registos com a finalidade de criar um grupo de potenciais registos duplicados. Em seguida cada registo no grupo é comparado com o registo de origem usando métodos de combinação de pares de registos. Segundo os autores de [25], um conjunto de registos que referem a mesma entidade pode ser interpretado de duas formas. Uma consiste em ver um dos registos do grupo como o correto e os restantes como duplicados que incluem informação com erros, sendo o objetivo a limpeza da base de dados. A outra forma, e a que foi adotada no projeto, consiste em considerar cada registo do grupo como uma fonte parcial de informação, sendo o objetivo juntar ou conciliar os registos duplicados gerando um único registo com mais e completa informação. No mesmo artigo, é ainda apresentada uma estratégia eficiente para identificar grupos de registos ou chamados *clusters*, que são aproximadamente duplicados, designado por algoritmo Union-Find, o qual foi tido em conta no desenvolvimento da solução.

Algoritmo Union-Find

O algoritmo Union-Find é bastante usado em implementações de Unificação de alto desempenho permitindo o acompanhamento dos conjuntos de registos de forma incremental. Utiliza uma estrutura de dados de conjuntos disjuntos, onde cada conjunto é identificado por um membro representativo do conjunto. Uma estrutura de dados deste género permite assim armazenar um conjunto de elementos em subconjuntos disjuntos, ou seja, que não se sobrepõem. A forma de representar os conjuntos é selecionar um elemento fixo de cada conjunto, chamado elemento representativo, para representar o conjunto como um todo.

Este algoritmo realizar três operações básicas sobre a estrutura de dados:

- Union (*x*,*y*) – Junta o conjunto que contém *x* e o conjunto que contém *y*, *Sx* e *Sy* respetivamente, num novo conjunto que é a sua união. Um representativo para a união é escolhido, e o novo conjunto substitui *Sx* e *Sy* na coleção de conjuntos disjuntos.

- Find (x) – Encontra a que conjunto x pertence, devolvendo o respetivo elemento representativo. Pode também ser usado para determinar se dois elementos estão no mesmo conjunto.
- MakeSet (x) – Permite criar um conjunto contendo um único elemento apenas.

Com estas três operações muitos problemas de particionamento podem ser resolvidos. A estrutura de dados usada no algoritmo Union-Find pode ser baseada em listas ou baseada em árvores [26].

Numa estrutura baseada em listas, um conjunto é representado por uma lista ligada. Cada elemento possui um apontador para o próximo elemento da lista assim como um apontador diretamente para a cabeça da lista, sendo este o elemento representativo do conjunto. Para juntar dois elementos basta juntar as listas que contêm cada um dos elementos, acrescentando uma lista a outra. E para comparar dois elementos apenas é necessário verificar se os dois elementos estão na mesma lista. Esta estrutura demonstrou ter um tempo de execução muito rápido.

Por outro lado, numa estrutura baseada em árvores, são utilizadas árvores com raízes, em vez de listas, para representar os conjuntos. Desta forma, a estrutura se assemelha a uma floresta de árvores, e cada nó da árvore contém um elemento e um apontador para o seu nó pai, se existir. Se dois elementos estão na mesma árvore, e uma vez que o elemento representativo consiste na raiz de uma árvore, então eles são considerados iguais, se os dois elementos têm a mesma raiz. Para juntar duas árvores, basta escolher um dos elementos representativos e definir o seu pai para ser o da outra.

A abordagem baseada em árvores não é melhor nem mais rápida do que a baseada em listas porque as árvores criadas podem ser altamente desequilibradas, comprometendo a sua eficácia e eficiência.

Descrição do método para deteção de identidades duplicadas

Uma vez que estamos a lidar com uma grande quantidade de dados em memória tem de ser feita uma boa gestão da mesma de modo a alcançar e manter uma boa eficácia e eficiência durante todo o processamento. Outro fator que é necessário ter em consideração é o custo computacional necessário para uma comparação individual entre dois registos, em que cada comparação pode requerer múltiplas comparações de atributos e cada comparação pode ser dispendiosa.

Para resolver estes problemas, a estratégia adotada para implementar o método para deteção de duplicados foi baseada nos princípios do algoritmo Union-Find (não sendo necessário implementar as três operações básicas deste) incorporando técnicas que melhoram em muito a velocidade de deteção. Ao seguir os princípios do Union-Find, os

registos que são aproximadamente duplicados ou que referem a mesma entidade são agrupados num *cluster* baseado em listas em que o primeiro elemento (cabeça da lista) diz respeito ao elemento representativo do grupo. Cada registo do grupo é considerado uma fonte parcial de informação e a junção de todos os duplicados sintetiza um registo mais rico em informação, sendo interpretado como o elemento representativo de todo o grupo. Assim, consegue-se obter uma identidade mais completa e consistente juntando toda a informação existente de várias fontes de dados dos respetivos duplicados do grupo. A estrutura de dados disjunta Union-Find permite gerir todos os *clusters* que vão sendo detetados e mantidos em memória.

Neste momento, sabemos como representar um conjunto de registos que referem uma mesma entidade, mas falta delinear uma forma para reduzir o número de comparações de pares de registos a fim de identificar os *clusters* para posteriormente os combinar. A estratégia usada para tal designa-se por *blocking* (bloqueio). Esta estratégia permite identificar duplicados analisando a tabela e calculando o valor de uma função de *hash* para cada registo. Este valor define o *bucket* (bloco) que é atribuído a um registo, e caso dois registos sejam duplicados então estarão atribuídos ao mesmo bloco. Para localizar e combinar duplicados basta comparar apenas os registos que estão num mesmo bloco ou *cluster*. Os blocos são subconjuntos mutuamente exclusivos, não existindo assim duplicados em diferentes blocos.

No enquadramento do projeto, a técnica de *hashing* não pode ser usada diretamente para aproximar duplicados, uma vez que não há garantia de que o valor de *hash* de dois registos similares será o mesmo. Assim, como alternativa ao uso do *hash*, o processo de deteção é executado em múltiplas passagens utilizando um critério diferente de bloqueio utilizando um atributo, de cada vez. Com isto, consegue-se reduzir substancialmente a probabilidade de falsas incompatibilidades com um pequeno aumento do tempo de execução. Assim, cada passagem de bloqueio corresponde a uma ordem de classificação diferente dos registos e é produzido um conjunto de pares de registos que podem ser combinados. No total, são realizadas oito passagens de deteção de duplicados utilizando oito atributos diferentes para o bloqueio. Após a computação destas passagens é executado o método de unificação de registos, descrito na secção seguinte, onde é utilizado um método de computação intensiva para comparar registos que possuem semelhanças mas que não têm campos necessariamente idênticos.

Para criar os *buckets* ou *clusters* de identidades foi usado um conjunto de atributos altamente discriminatórios para efetuar o bloqueio, sendo a ordem usada para realizar as passagens a apresentada e os atributos os seguintes:

1. Id_NM;
2. Id_NC;
3. Id_BI;
4. Id_PA;
5. Id_AR;
6. AD_Users;
7. AD_Emails;
8. FullName.

A deteção de duplicados é então baseada em vários atributos para que de seguida seja realizada a unificação desses registos segundo esses mesmos atributos. Isto, porque existem registos que, por exemplo, não têm valores de Id_NM mas sim de Id_BI. E desta forma, é possível unificar o máximo número de registos possível. A forma de realizar o bloqueio consiste num método determinista, ou seja, dois registos A e B estão num mesmo bloco ou *cluster* de ID_NM se os valores de ID_NM de ambos os registos são idênticos, o mesmo acontece para os restantes atributos discriminatórios. A similaridade não foi usada, pois é muito pesada computacionalmente existindo mais comparações a realizar entre *Strings* usando a métrica Jaro Winkler-Distance, logo os registos de nome "rosa" e "roza" não vão pertencer ao mesmo *cluster*.

De entre o conjunto de atributos discriminatórios, a unificação segundo o atributo Id_BI é aquele que carece de um aspeto importante. Baseado no estudo realizado aos números de identificação utilizados pela organização (descrito no subcapítulo 2.3), constatou-se que um mesmo número de Bilhete de Identidade pode ser representado de duas diferentes formas. Por exemplo, os números "00123456" e "123456" referem o mesmo número de Bilhete de Identidade. O primeiro valor contém zeros à esquerda até perfazer um total de 8 dígitos, enquanto, que o segundo valor excluiu os zeros à esquerda. Desta forma, a unificação de registos segundo este atributo teve este aspeto em consideração no processo de sintetização de identidades aglomerando no mesmo *cluster* registos que possuam as duas representações possíveis (com e sem zeros).

Por último acontece a unificação pelo atributo FullName porque verificou-se que no final do processamento de unificação ainda existirem registos duplicados em que estes apenas podem ser combinadores através do nome, uma vez que possuem somente o nome e pouco mais como elementos de identificação para identificar a sua personalidade operacional. No entanto, ao usar uma unificação deste tipo poderão surgir falsos positivos quando dois nomes idênticos poderão na realidade referirem diferentes identidades especialmente se forem constituídos apenas pelo primeiro e último nome.

A ordem pela qual é efetuada a deteção segundo estes oito atributos dita também o peso associado a cada atributo. Primeiro, os registos são detetados segundo os

elementos de maior peso, ou seja, os cinco elementos de identificação, pois permitem detetar duplicados com uma forte relação, seguindo-se os elementos de menor peso, ou seja, aqueles que são mais suscetíveis a erros e menos credíveis podendo surgir falsas compatibilidades de combinações, como por exemplo, no atributo AD_Emails é muito comum ser usado um endereço de *email* da pessoa que autorizou a abertura de conta e não o do respetivo utilizador da conta.

Neste momento, apenas é necessário desenvolver e implementar uma estrutura de dados que represente todos os aspetos anteriores para que consigamos obter resultados de forma precisa e eficiente. Para tal, a estrutura de dados escolhida para representar uma passagem de bloqueio foi uma *inverted list* ou *inverted index* (lista ou índice invertido), existindo uma para cada critério de bloqueio. Desta forma, o *cluster* ou *bucket* de registos é enumerado rapidamente. Um índice invertido é considerado um tipo de organização de elementos e uma estrutura de dados que mapeia elementos de acordo com as suas ocorrências, por exemplo, armazenados em bases de dados ou ficheiros. É considerado também uma lista ordenada de chaves em que cada chave contém uma ligação para os documentos que a contêm, sendo portanto uma lista de chaves primárias associadas a uma chave secundária. Estas estruturas nasceram na necessidade de pesquisas por termos numa lista tradicional exigirem percorrer cada documento e palavra dentro destes em busca do termo, e usar índices invertidos permitiu saltar diretamente para o termo pesquisado, melhorando a eficiência das pesquisas. Este mecanismo permitiu assim obter resultados de forma consideravelmente mais rápida e precisa. É chamado por inverter a hierarquia da informação, em vez de uma lista de documentos contendo termos, é obtida uma lista de termos, referenciando documentos (através de um identificador único, como uma chave primária).

Desta forma, a estratégia de indexação adotada recorreu ao uso de um índice invertido por este se demonstrar muito vantajoso quando se lida com grandes quantidades de registos, possibilitante pesquisas bastante rápidas. Como cada passagem de bloqueio é independente e como existem oito passagens no total vão existir oito índices invertidos. Cada índice mapeia um par de valores (chave/valor) associando uma chave a cada *cluster* ou *bucker* criado. O *cluster* criado é representado por um objeto da classe do atributo de bloqueio que contém um atributo “recs” que consiste numa lista destinada a guardar as identidades cujo utilizam a respetiva chave, como veremos mais à frente. Por exemplo, se considerarmos a passagem segundo o atributo Id_NM então a chave desta estrutura seria um valor de Id_MN e o seu *cluster* armazenaria o grupo de registos que utilizasse esse valor de Id_NM, por meio de um objeto da classe NM. No índice invertido segundo o atributo FullName, o mapeamento das chaves é realizado após colocação no formato canónico de cada *string*, procedendo à remoção de acentos e carateres especiais.

Para simplificar e exemplificar a utilização destes índices é apresentado o método de deteção de registos duplicados aplicado a oito registos que representam identidades, indicando o estado de algumas passagens de bloqueio e respetivos índices invertidos.

Lista de identidades = | R1 | R2 | R3 | R4 | R5 | R6 | R7 | R8 |

	FullName	Id_NM	Id_BI	AD_Users	AD_Emails	Company_PT
R1	João Ferreira	10012345	-	xpt1	-	PT Contact
R2	João F.	10012345	12345678	xpt1	-	PT Contact
R3	João	10012345	-	xpt1	-	PTSI
R4	Maria Joana	-	11112222	xpt2	maria@telecom.pt	-
R5	Joana	-	-	-	maria@telecom.pt	-
R6	Pedro	10067898	-	xppp	pedro@telecom.pt	PTSI
R7	Luís	10072134	-	yxpl	-	PTSI
R8	Ana	10012349	-	xmlp10	-	PTSI

Tabela 4.3 - Lista de registos de identidades

Após a execução do método de deteção de duplicados, o conteúdo dos seguintes índices invertidos é:

Chave	Valor
10012345	{R1,R2,R3}
10067898	{R6}
10072134	{R7}
10012349	{R8}

Tabela 4.4 – Índice de Id_NM

Chave	Valor
12345678	{R2}
11112222	{R4}

Tabela 4.5 - Índice de Id_BI

Chave	Valor
xpt1	{R1,R2,R3}
xpt2	{R4}
xppp	{R6}
yxpl	{R7}
xml10	{R8}

Tabela 4.6 - Índice de AD_Users

Chave	Valor
maria@telecom.pt	{R4,R5}
pedro@telecom.pt	{R6}

Tabela 4.7 - Índice de AD_Emails

Neste cenário, os índices de Id_BN, Id_PA e Id_AR estariam vazios pois não existem quaisquer valores.

Como referido anteriormente, uma vez que é necessário armazenar mais informação acerca dos valores das chaves de cada índice foi desenvolvida a superclasse de objetos UserID que contém as classes NM, NC, BI, PA, AR, ADUsers, ADEmails e FullName, descritas no subcapítulo 4.2. Cada uma destas subclasses representa um valor de cada chave usada nos índices e adicionalmente um conjunto de campo úteis a todo o processamento, incluindo o atributo "recs" que se destina a armazenar o *cluster* (conjunto de duplicados), sob a forma de objetos da classe Identity.

Segundo o diagrama de classes UML, no início do processo de deteção cada registo lido da tabela SourceAccount representa um objeto SourceAccount. De seguida, é transformado numa personalidade mestre ou objeto Identity e numa personalidade operacional ou objeto Account (apontando para o objeto Identity), quando se trata de registos de contas aplicacionais. Nos registos provenientes das restantes fontes de dados (OIM e Pulso/Acessos) apenas interessa a personalidade mestre.

Para exemplificar a utilização de índices invertidos aplicado ao procedimento de deteção de duplicados segundo o paradigma orientado a objetos, os objetos aplicados ao conjunto de registos anteriores originam inicialmente oito objetos Identity com os respetivos atributos. De seguida, ao aplicar este método de deteção de duplicados apenas ao atributo de bloqueio Id_NM, o estado deste índice seria da seguinte forma:

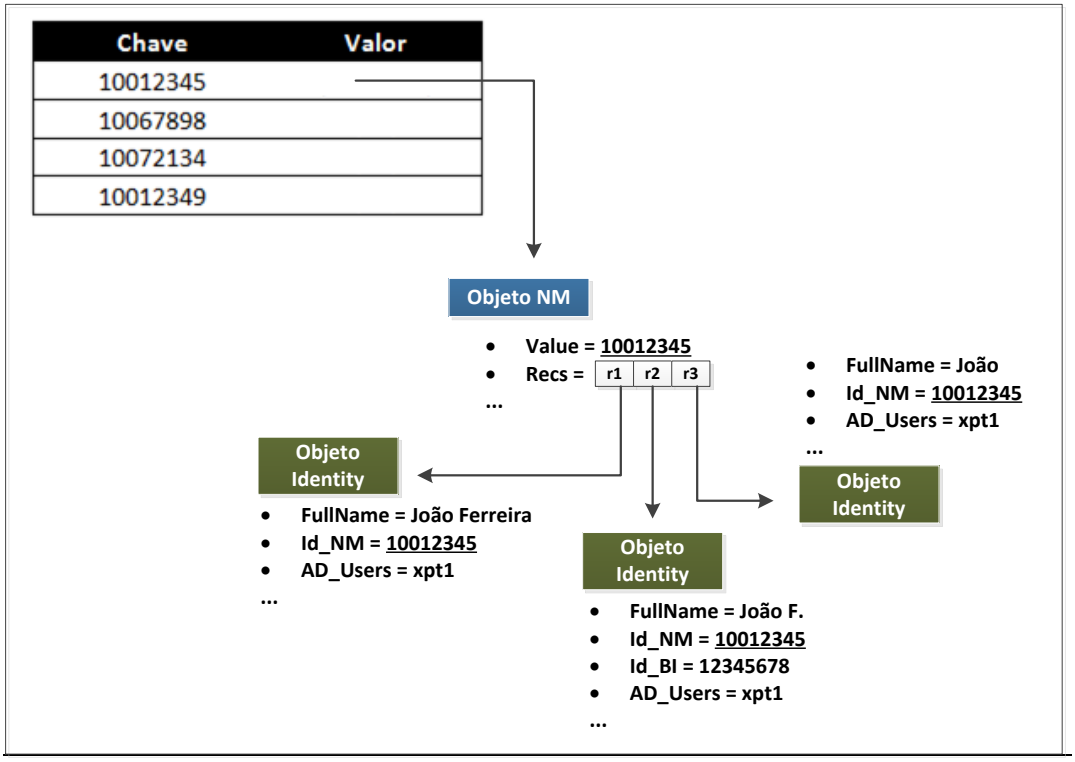

Figura 4.6 – Estado do índice de Id_NM após deteção de duplicados

Nos índices invertidos que incidem sobre os números identificativos não existe o mesmo objeto Identity em várias entradas do mesmo índice. Isto, porque uma identidade não pode ter múltiplos valores destes tipos de atributos. Nos índices de Ad_Users e Ad_Emails podem existir diferentes valores a serem usados pelo mesmo objeto Identity, porque uma identidade pode ter vários valores deste tipo de atributos. Com a utilização de índices invertidos, o tempo de computação reduziu extremamente em comparação com o método naïve na comparação de registos devido à gestão das estruturas de dados Union-Find, *clusters*. A complexidade temporal reduziu de $O(n^2)$ para $O(n \,.\log n)$, sendo *n* o total de registos.

Para concluir, esta fase de deteção de duplicados consiste em fazer múltiplas passagens de bloqueio e agrupamento, em que cada passagem os *clusters* são atualizados de acordo com um atributo ou critério de classificação de domínio independente. Cada passagem representa uma unificação segundo um critério ou atributo diferente. O objetivo desta fase não é identificar pares de registos que combinam ou não, mas sim aglomerar e detetar registos que potencialmente podem ser alvo combinações.

Na próxima secção, é apresentado o método de unificação de registos que baseado no método de deteção de duplicados e num conjunto de regras consegue-se proceder à sintetização das identidades realizando comparações de pares de registos num *cluster*, incidindo sempre sobre o elemento representativo do grupo e um duplicado.

4.4.2 Método para combinação de registos duplicados

Após realizar a deteção de registos ou identidades duplicadas, os índices invertidos são atualizados referindo cada instância de valor encontrado associado a um *cluster* de registos duplicados. Estes grupos de registos são na realidade objetos da classe Identity, segundo o diagrama de classes em 4.2. De seguida, e para concluir o processo de sintetização de identidades segue-se o procedimento para combinar ou unificar pares de registos identificados como duplicados nos *clusters*.

No desenvolvimento deste método foram testadas duas metodologias diferentes para unificação dos *clusters*. Uma metodologia consistiu em aplicar os métodos de deteção e combinação de todos os registos presentes na tabela Source_Accounts de uma só vez, mas este processo demonstrou ser bastante caro computacionalmente levando a um aumento do tempo de processamento. Em alternativa, e a qual foi escolhida consistiu em aplicar os mesmos métodos mas de forma individual aos registos de Source_Account a cada fonte de dados ou rotina de ETL. Com a adoção desta metodologia, o processo de sintetização ficou mais eficiente sendo processados menos objetos e contribuindo para uma melhor gestão da memória.

Também foi possível unificar identidades segundo uma determinada ordem, sendo esta fundamental quando estão envolvidas diferentes fontes de dados caracterizadas por diferentes elementos de informação. Com a definição desta ordem, parte-se da fonte de dados com mais informação para a que contém menos com o objetivo de aumentar a probabilidade de unificar o número máximo de identidades, ou seja, obter o número máximo de relações entre as personalidades operacionais e personalidades mestre.

É importante salientar que os registos obtidos da tabela temporária Source_Accounts são obtidos de forma ordenada, usando a seguinte cláusula SQL:

```
    SELECT * FROM InfoTemp.Source_Accounts

      ORDER BY id_nm, id_nc, id_bi, id_pa, id_ar, ad_user, ad_email,
fullname ASC
```

Ordem de combinação de fontes de dados

Segundo o processo de ETL descrito no subcapítulo 4.3 existem 24 rotinas que resultarão em 24 ciclos ou iterações de unificação. Como referido anteriormente, a ordem pela qual é feita a unificação é muito importante, uma vez que, queremos começar o processo de sintetização pela fonte de dados da plataforma OIM (e respetivas fontes de dados OIM de cada aplicação). Estas fontes referem informação acerca de identidades que abriram conta e existe essa identidade na plataforma OIM, onde já existe um conjunto de identidades sintetizadas. De seguida, é usada a fonte da AD onde esta contém informações acerca de contas de rede dos utilizadores, sendo muito importante porque além de possuir muitos elementos informativos acerca da personalidade operacional também grande parte das aplicações autenticam na AD, ou seja, utilizam o nome de utilizador de rede para aceder à aplicação. Posteriormente são tratadas as fontes das próprias aplicações, em que a ordem destas fontes começa naquela com mais informação terminando naquela com menos informação. Por último, e apenas para complementar as identidades sintetizadas até ao momento são utilizadas as fontes dos Pulso/Acessos das mesmas aplicações, não sendo criadas neste passo novas identidades a partir destas fontes de dados. Este conjunto de fontes de dados refere pedidos de abertura de contas em que um pedido concluído pode indicar informação acerca de um individuo que já não tem essa mesma conta que pediu acesso nem outras contas, e pode também já não existir na plataforma OIM.

Para facilitar a definição desta ordem de unificação foi construído um conjunto de grupos de fontes de dados:

- Grupo 1: Fontes de dados do OIM.
- Grupo 2: Fontes de dados da AD e de aplicações.

- Grupo 3: Fontes de dados do Pulso/Acesso.

Cada unificação de uma fonte de dados é realizada segundo o vetor de atributos determinísticos anteriormente indicado, ou seja, Id_NM, Id_NC, Id_BI, Id_PA, Id_AR, AD_Users, AD_Emails, FullName (e pela mesma ordem). Portanto, para cada fonte de dados processada através da respetiva rotina de ETL é feita a unificação com base neste conjunto de atributos. A unificação segundo esse vetor começa por tratar primeiro os colaboradores internos e posteriormente os externos, uma vez que, estes geralmente apresentam mais elementos de informação. Um colaborador é interno se possuir número mecanográfico, caso contrário é considerado colaborador externo.

Como mencionado anteriormente, todas as fontes de dados envolvidas possuem uma componente identidade, e adicionalmente, apenas as fontes de dados das aplicações possuem uma componente conta. Desta forma, o processo de sintetização envolve apenas as componentes identidades para efeitos de unificação.

Baseado no conjunto de todas as contas aplicacionais sob forma canónica é esperado que cerca de 90% destas contas satisfaçam ou unifiquem com as identidades da plataforma OIM e os restantes 10% não unifiquem mas permitem criar novas identidades, expandindo ou complementando assim as identidades atuais da plataforma OIM, enriquecendo esta plataforma. Esta pequena percentagem de novas identidades pode derivar de registos "perdidos" ou isolados, isto é, que não têm um dos cinco números identificativos obrigatórios mas sim apenas o nome e pouco mais para identificar essas personalidades. Esses registos podem se tratar de contas em que a sua abertura que não passa atualmente pela plataforma Pulso/Acesso (e OIM) e mesmo assim existe informação necessária suficiente para ser criada uma nova identidade, ou, podem-se tratar de contas isoladas não existindo informação suficiente que caracterize o proprietário da conta. A longo prazo espera-se que estes 10% diminuam.

Para combinar um par de registos é necessário estabelecer previamente um conjunto de regras que indique se dois registos estão ou não aptos para serem unificados, assim como um conjunto de regras para combinação dos seus atributos, como veremos de seguida.

Regras de combinação

As regras de combinação implementadas são verificadas aquando da combinação ou unificação de duas identidades e estão divididas em dois grupos: combinação de identidades e combinação de atributos.

Regras de combinação de identidades

Como vimos anteriormente, a combinação de duplicados incide exclusivamente sobre o elemento representativo do *cluster* e um dos duplicados, e portanto todas as atualizações de atributos são feitas sobre este. Também, durante a combinação de registos são verificadas um conjunto de regras antes e depois da unificação propriamente dita. Estas heurísticas permitem assim acelerar o processo de comparação de registos e manter a qualidade dos atributos que constituem uma identidade.

As regras verificadas previamente são extremamente importantes e caso sejam violadas anulam a unificação desfazendo todos os passos efetuados até ao momento. Ainda, as duas identidades são "marcadas" como *tainted* (estragadas), assim como também o atributo pelo qual foi desencadeada a unificação. As regras consideradas graves são:

- Quando existem ou estão instanciados diferentes valores de Id_NM, Id_NC, Id_BI, Id_PA ou Id_AR.
- Quando existem nomes completos muito distantes ou diferentes, recorrendo à técnica de verificação de similaridade usando a métrica de distância Jaro-Winkler Distance. Devido a observar-se em determinadas fontes de dados nomes constituídos somente por carateres numéricos, estes são compreendidos como vazios para ser possível a combinação das duas identidades, e não resultar numa falha de unificação, uma vez que se tratam de nomes inválidos.

No entanto, devido à baixa credibilidade e susceptibilidade a erros da combinação segundo o atributo FullName, durante a falha de unificação não serão marcadas como *tainted* as identidades e o atributo envolventes. Adicionalmente, nesta combinação de duplicados não se torna necessário verificar a segunda regra acima, pois todos os registos dos *clusters* obedecem a essa regra.

As regras verificadas posteriormente não desfazem a unificação nem marcam as identidades e o atributo como *tainted*, e como são aplicadas no fim incidem somente sobre o elemento representativo do *cluster*. As regras não consideradas graves são:

- Ausência de pelo menos um valor de Id_NM, Id_BI, Id_NC, Id_PA ou Id_AR, que identifica a identidade.
- Ausência de um nome de utilizador.
- Existência de diferentes endereços de *email* de domínio Telecom.
- Existência de diferentes valores de utilizadores de rede.
- Quando o nome usado no endereço de *email* é diferente do nome completo.

Em qualquer um dos tipos, quando uma regra não é satisfeita é emitida uma mensagem de aviso que fica armazenado no campo de anotações das identidades envolvidas. Para representar estas mensagens optou-se pela utilização de códigos, sendo eles os seguintes:

Código	Extensão	Significado
FDN	Fail Different Names	Diferentes nomes completos
FDI	Fail Different Ids	Diferentes valores de identificação (NM, NC, BI, PA ou AR)
DNE	Different Name-Email	Diferente nome usado em FullName e Ad_Emails
ME	Many Ad_Emails	Demasiados endereços de *email* de domínio Telecom
MU	Many Ad_Users	Demasiados utilizadores de rede

Tabela 4.8 - Tabela de códigos de unificação

Para efeitos de qualidade de dados, uma identidade não deverá possuir mais do que um endereço de *email* corporativo (de domínio Telecom) e não deverá possuir mais do que um utilizador de rede.

Os códigos FDN e FDI são bastante úteis neste procedimento graças ao formato escolhido, possibilitando ao analista de segurança identificar anomalias no processo de unificação. Estes códigos surgem durante uma falha de unificação entre duas identidades, e o formato adotado foi o seguinte:

FDN/FDI (identity_id, atributo)

O parâmetro "identity_id" indica o identificador da identidade que falhou a unificação com a identidade corrente e o parâmetro "atributo" indica o tipo de atributo que despoletou a unificação. O valor desse atributo é comum a ambas as identidades envolvidas na combinação.

É importante referir que quando são comparados dois valores para verificação destas regras são previamente colocados os valores no seu formato canónico (descrito no subcapítulo 4.1).

Regras de combinação de atributos

Estas regras são utilizadas para combinar valores de atributos quando se pretende combinar um par de identidades. Para descrever estas regras são considerados a título de exemplo dois registos A e B, e dois valores X e Y, respetivamente, de um mesmo atributo. Sendo A o registo elemento representativo do grupo, e tal como foi dito acima, o resultado da combinação de atributos vai incidir sobre este registo, e B um duplicado de A.

As regras gerais de combinação são: concatenação de Y a X se ambos forem diferentes entre si, ou, obtenção de valor Y se X for vazio ou não instanciado. A maioria

dos atributos utiliza esta convenção mas, no entanto, existem alguns atributos que requerem a aplicação de regras mais específicas, tais como:

- FullName – escolhido o nome mais longo.
- Id_NM, Id_NC, Id_BI, Id_PA e Id_AR – escolhido Y caso X seja vazio (estes campos nunca agregam mais do que um único valor).
- AD_Users_Num e AD_Emails_Num – são calculados em função do número de valores de AD_Users e AD_Emails.
- Company_PT – tem o valor "PTP" (PT Portugal) se X e Y refiram nomes de empresas diferentes, caso contrário são concatenados.
- Activity_Level e Accum_Risk_Level – somado Y a X.
- Matches_Num – é adicionado Y a X e incrementada 1 unidade.

Além destes atributos, existem outros que não devem ser alvo de atualizações aquando da combinação de registos, como por exemplo, Identity_Id, Tainted, Fail_Ids, Source_ETL, Identity_Type, entre outros.

Durante a combinação de atributos resultam atributos multivalor, ou seja, que possuem mais do que um valor, e em qualquer um dos casos nunca são repetidos valores.

Descrição do método de combinação de identidades duplicadas

Após o preenchimento de todos os índices invertidos segue-se a combinação dos duplicados presentes nos vários *clusters* de cada um dos índices invertidos, comparando pares de identidades. Considerando um grupo de duplicados, as comparações são sempre realizadas sobre o elemento representativo do grupo e um dos duplicados, até estes serem esgotados. Com isto, consegue-se obter no fim uma identidade mais completa e rica em informação.

Para combinar um par de identidades combinam-se os valores dos vários atributos, de acordo com as regras definidas acima. Considerando um par de identidades A e B:

- Existe combinação de B com A se não for violada uma regra grave, e portanto prossegue-se a:
  - Combinação de valores de atributos de B com A segundo as regras de combinação de atributos anteriormente descritas;
  - Atualização do atributo Identity_Id das contas pertencentes a B para estas "apontarem" para A;
  - Atualização do atributo Confidence_Level das contas pertencentes a B;

- Remoção da identidade B e atualização dos índices invertidos substituindo a identidade B pela identidade A.

- Não existe combinação de B com A se for violada uma regra grave, e portanto prossegue-se a:
  - Marcação das duas identidades e o valor do atributo pelo qual iniciou a unificação como *tainted*;
  - Escrita no atributo Annotations das duas identidades indicando o motivo da falha de unificação, os identificadores de ambas as identidades e o valor do atributo de unificação.

Para concluir, o processo de sintetização foi desenvolvido de forma flexível para pode ser aplicado a qualquer quantidade de registos de entrada e de registos duplicados, permanecendo de forma inalterada. De seguida, e por fim, será apresentado um conjunto de casos de testes usados para exemplificar os passos efetuados neste processo.

Foram criadas duas bibliotecas de funções que auxiliam o processo de sintetização de identidades, "Clustering_Functions" e "Clustering_Rules".

4.4.3 Exemplos de sintetização de duplicados

Para concluir a descrição deste processo são apresentados a título de exemplo vários grupos de registos duplicados caracterizados por diferentes elementos de informação submetidos ao método de combinação descrito. Como vimos, a ordem segundo os atributos pela qual é realizada a unificação é a seguinte:

Id_NM	Id_NC	Id_BI	Id_PA	Id_AR	AD_Users	AD_Emails	FullName

Tabela 4.9 - Ordem de unificação segundo atributos

De forma a simplificar a visualização dos exemplos neste documento foram abreviados alguns nomes de atributos referentes a uma identidade:

a) Exemplo 1

FullName	Id_NM	Id_NC	Users	Domains	Emails	Cmp_PT
Joao Ferreira	10011133	-	-	-	-	-
JOAO FERREIRA	10011133	250100200	T011222	PTCOM	joao-ferreira @telecom.pt	PT Contact
João Ferreira	-	-	T011222	-	joao-ferreira @telecom.pt	-

Tabela 4.10 – Exemplo 1 de sintetização (antes)

FullName	Id_NM	Id_NC	Users	Domains	Emails	Cmp_PT
João Ferreira	10011133	250100200	T011222	PTCOM	joao-ferreira @telecom.pt	PT Contact

Tabela 4.11 - Exemplo 1 de sintetização (depois)

A sintetização dos três duplicados resultou num único e mais completo registo. As unificações realizadas foram, por ordem, segundo os atributos Id_NM e AD_Users. As unificações segundo os atributos AD_Emails e FullName não foram necessárias, uma vez que as anteriores esgotaram os duplicados existentes.

b) Exemplo 2

FullName	Id_NM	Id_NC	Id_BI	Users	Emails	Cmp_PT
LUIS SANTOS	-	-	-	TM01333	pedro.alexandre @tmn.pt	-
LUIS SANTOS	10001221	-		-	-	-
Luis Santos	10001221	250100400	12233567	tm01333	p-alves @telecom.pt	TMN
LUIS SANTOS	10013133	-	-	-	luis-m-santos @telecom.pt	TMN
LUIS SANTOS	10013133	-	-	-	-	-
LUIS SANTOS	10013133	251400300	17653322	U08888	luis-m-santos @telecom.pt	PT Comunicacoes

Tabela 4.12 - Exemplo 2 de sintetização (antes)

FullName	Id_NM	Id_NC	Id_BI	Users	Emails	Cmp_PT
LUIS SANTOS	10001221	250100400	12233567	TM01333	pedro.alexandre @tmn.pt, p-alves @telecom.pt	TMN
LUIS SANTOS	10013133	251400300	17653322	U08888	luis-m-santos @telecom.pt	PTP

Tabela 4.13 - Exemplo 2 de sintetização (depois)

A sintetização dos seis duplicados iniciais resultou em dois mais completos registos. As unificações realizadas foram, por ordem, segundo os atributos Id_NM (várias iterações) e AD_Users. A unificação segundo o atributo AD_Email não foi necessária, uma vez que as anteriores esgotaram os duplicados existentes. A unificação segundo o atributo FullName não é efetuada devido à existência de diferentes valores de Id_NM, Id_NC e Id_BI, violando assim a respetiva regra e assim é "descartada".

c) Exemplo 3

FullName	Id_NM	Id_NC	Id_BI	Users	Emails	Cmp_PT
Joao Miguel Ferreira	-	-	-	tm01414	Joao-m @telecom.pt	PTCOM
Joao Miguel Ferreira	10001414	-	-	-	-	-
Joao Ferreira	10001414	222333555	58321	tm01414	Joao-m @telecom.pt	TMN

Tabela 4.14 - Exemplo 3 de sintetização (antes)

FullName	Id_NM	Id_NC	Id_BI	Users	Emails	Cmp_PT
Joao Miguel Ferreira	10001414	222333555	58321	tm01414	Joao-m @telecom.pt	PTP

Tabela 4.15 - Exemplo 3 de sintetização (depois)

A unificação foi realizada inicialmente segundo o índice invertido do atributo Id_NM, e de seguida pelo do atributo AD_Users. As unificações segundo os atributos AD_Emails e FullName não chegaram a ser concretizadas porque os registos foram esgotados nas anteriores unificações.

d) Exemplo 4

Identidade	Id_NM	Id_NC	Id_BI	AD_Users
A	10012345	-	10000099	XPA
B	10012345	11112222	-	XPB,XPC
C	111345	-	12345	XXX
D	-	-	12345	AAA

Tabela 4.16 - Conjunto de identidades iniciais

Antes de qualquer unificação, o estado dos seguintes índices invertidos é:

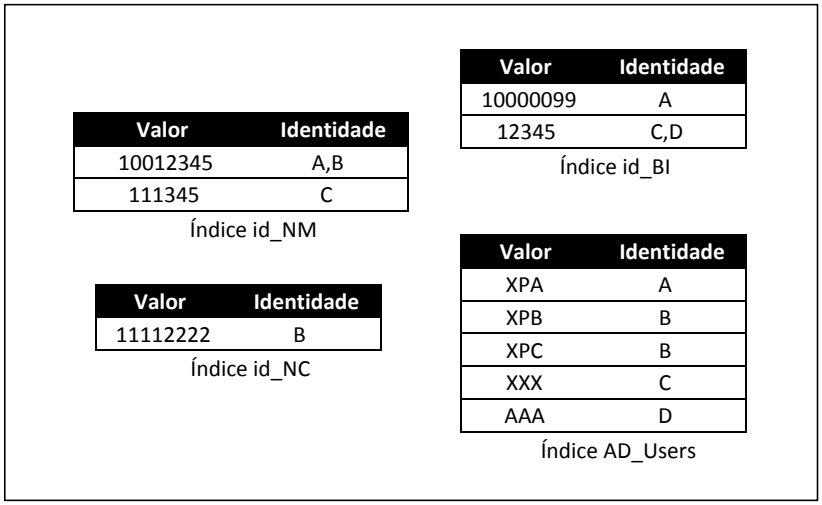

Figura 4.7 - Estado dos índices invertidos (antes de qualquer unificação)

Os restantes índices invertidos estão vazios. Segundo a ordem de unificação mencionada, a primeira tentativa de unificar estas identidades inicia-se segundo o índice do atributo Id_NM, resultando no seguinte estado:

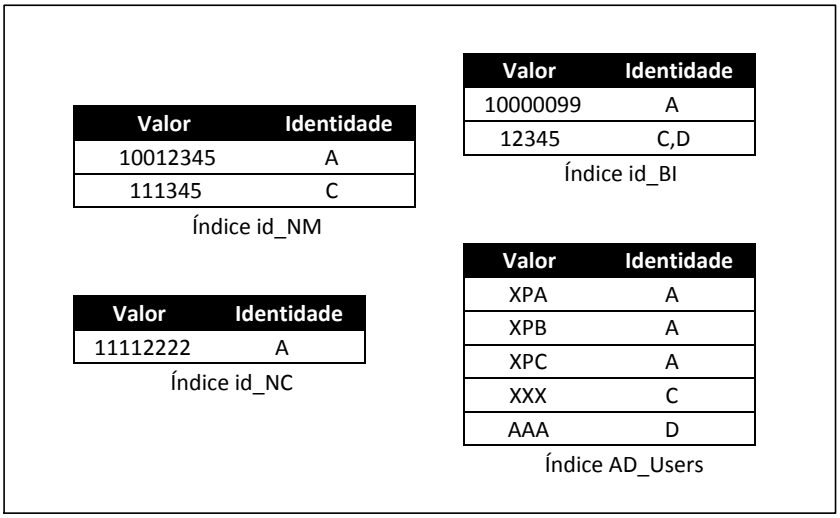

Figura 4.8 - Estado dos índices invertidos (após unificação por Id_NM)

Por último é realizada a unificação segundo o atributo Id_BI, resultando no seguinte estado:

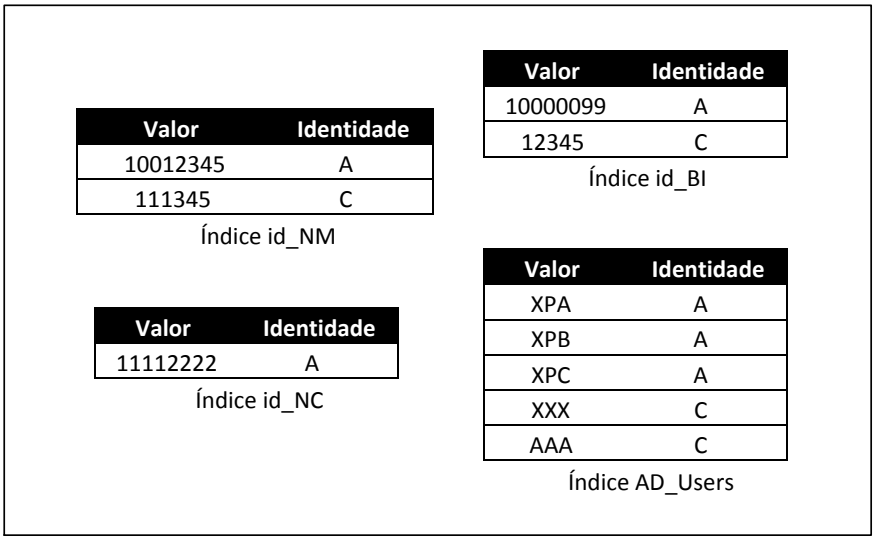

Figura 4.9 - Estado dos índices invertidos (após unificação por Id_BI)

Por fim, as identidades sintetizadas são:

Identidade	Id_NM	Id_NC	Id_BI	AD_Users
A	10012345	11112222	10000099	XPA,XPB,XPC
C	111345	-	12345	XXX,AAA

Tabela 4.17 - Conjunto de identidades sintetizadas

4.5 Processo de validação de dados

De forma a melhorar a qualidade dos dados envolvidos foi desenvolvido uma biblioteca de operações de validação de dados com o objetivo de classificar a estrutura e estado de validade dos valores de origem, designado por DataValidator. A validação é realizada sobre os dados finais após o processamento de sintetização das identidades, e incide sobre valores de números de Bilhetes de Identidade ou Cartão de Cidadão, números de Contribuinte ou de Identificação Fiscal, números Mecanográficos, endereços de *email* e datas.

Os algoritmos implementados nas operações são baseados na análise de números de identificação realizada no subcapítulo 2.3, e resultam num código de erro associado ao valor em questão, podendo estar relacionado com a sua estrutura ou com o seu estado válido. Inicialmente foi utilizado um sistema de códigos baseados em números inteiros mas uma vez que o número de códigos não tende a ser muito extensa e devido a não serem muito percetíveis e intuitivos, foram utilizadas letras em vez de números. Foi então necessário definir uma lista de códigos de erros possíveis que podem surgir para esses atributos. Na seguinte lista podemos visualizar os códigos de erros, e quando um erro ocorre todos os seus antecessores não acontecem, ou seja, são falsos.

- Endereço de *email*
  - A – Endereço mal estruturado, ou seja, estrutura diferente do formato "nome@dominio" ou domínio utilizado não permitido pela organização.
  - B – Endereço não existente, verificado através de sucessivas chamadas ao servidor do domínio.
- Data
  - D – Data mal estruturada, estrutura diferente dos formatos padrão ou intervalo inválido.
- Número de Bilhete de Identidade
  - F – Valor não numérico.
  - G – Número sem dígito de controlo ou tamanho diferente de 8 dígitos.
  - H – Dígito de controlo errado.

Para validar um número de Bilhete de Identidade é necessária a existência do dígito de controlo ou 9 dígitos para a sua verificação.

- Número de Contribuinte
  - I – Valor não numérico.
  - J – Tamanho diferente de 9 dígitos.

- L – Valor não iniciado pelos dígitos 1,2,5,6,8 ou 8.
- M – Dígito de controlo errado.

- Número Mecanográfico
  - N – Valor não numérico.
  - O – Tamanho diferente de 8 dígitos.
  - P – Número fora dos limites da série ou número de série inválido.

Esta lista de erros permite assim melhorar a qualidade dos dados indicando de forma detalhada o erro causado, facilitando a sua correção numa fase posterior.

4.6 Plataforma PiasaServer

Com recurso às ferramentas listadas no subcapítulo 2.4 e com base nos processos apresentados anteriormente, nomeadamente, processo de ETL, processo de sintetização de personalidades mestre e processo de validação de dados, foi desenvolvida uma plataforma utilizando a linguagem Ruby que resolve os problemas analisados no subcapítulo 2.1 e alcançam os objetivos descritos no subcapítulo 1.2.

A plataforma designa-se por PiasaServer e utiliza a base de dados PiasaInfo para armazenar os resultados devidamente processados. Como o processamento do projeto consiste em operações bastante complexas, decidiu-se dividir a plataforma em três fases de processamento distintas, cada uma desempenhando um conjunto de tarefas específico, como veremos de seguida.

- PreClustering;
- Clustering;
- PosClustering.

4.6.1 Fase I – PreClustering

A fase de PreClustering é a etapa inicial de todo o processamento da solução desenvolvida. Nesta fase são realizadas operações de criação de uma nova instância ou sessão correspondendo a uma nova execução do processo de sintetização de identidades, assim como também operações de preparação dos dados que servem de suporte dados à fase seguinte. As sessões são catalogadas com base na data atual. Uma vez que a aplicação vai ser executada diariamente apenas uma vez ao dia, se esta for executada mais do que uma vez, a sessão anteriormente criada é substituída pela corrente. As operações realizadas nesta fase são:

- Preparação do sistema de armazenamento relacional;

- Preparação do sistema de ficheiros;
- Processamento das rotinas de ETL;

Operação 1: Preparação do sistema de armazenamento relacional

Operação de preparação de uma nova base de dados relacional e respetivas tabelas a utilizar na nova sessão, segundo o modelo de dados no subcapítulo 3.2. O nome da base de dados a utilizar como *output* de resultados segue o formato "piasa_info_[data_atual]", como por exemplo, "piasa_info_2013_09_01", que significa que no dia 01 de Setembro de 2013, o processo de sintetização arrancou e produziu dados na base de dados "piasa_info_2013_09_01". A base de dados temporária InfoTemp mantem o mesmo nome e o conteúdo da sua tabela Source_Accounts é apagado a cada sessão.

Esta operação faz limpar todo o conteúdo destas tabelas se já estiverem criadas. Esta forma adotada de criação de base de dados a cada execução é útil pois é sempre preservado a informação do dia anterior, protegendo assim a informação já sintetizada quando algum *crash* (interrupção) ocorra durante a execução da aplicação corrente. Assim como também é útil perceber a evolução das contas e identidades a longo prazo para efeitos estatísticos.

Operação 2: Preparação do sistema de ficheiros

Operação de preparação do sistema de pastas e ficheiros a utilizar pela sessão corrente. Cada sessão dispõe de uma pasta de ficheiros dedicada para armazenar todos os ficheiros que são produzidos durante o processo de sintetização.

A estrutura do sistema de ficheiros é a seguinte:

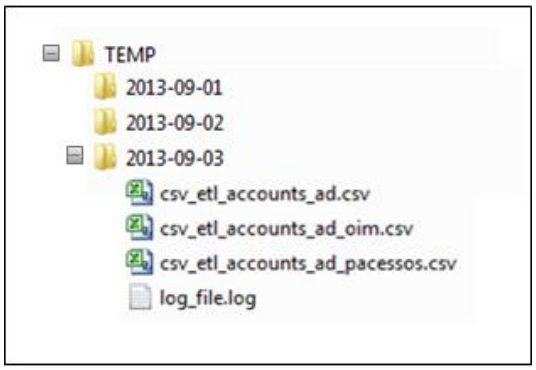

Figura 4.10 - Sistema de ficheiros

A pasta TEMP é a pasta principal ou raiz com o objetivo de guardar todas as sessões criadas. Se esta pasta não existir será criada. As subpastas equivalem a uma sessão e são denominadas pela data atual da criação da sessão, por exemplo, a pasta "2013-09-03" corresponde a uma execução da aplicação no dia 03 de Setembro de

2013, e no seu interior estão os ficheiros gerados por ela. Ao criar uma subpasta para uma sessão se esta já existir então todo o seu conteúdo é apagado.

Adicionalmente é utilizado um ficheiro de *log* "log_file.log" que vai guardando todas as mensagens de eventos ou operações e respetivos tempos de execução ocorridos durante a execução do processo de unificação. Um ficheiro deste tipo auxiliou bastante o desenvolvimento da aplicação sendo útil para identificação de eventuais anomalias. É utilizada a gema Ruby Logging[26] para escrever neste ficheiro que consiste numa biblioteca de funções que permitem escrever *logs* (mensagens) importantes de eventos de várias formas para diferentes destinos.

Operação 3: Processamento das rotinas de ETL

Operação de processamento ou execução das rotinas de ETL descritas no subcapítulo 4.3. A ordem pela qual são executadas as rotinas é irrelevante, pois neste momento apenas estamos interessados em converter cada conta de origem "normal" para uma conta canónica.

Como vimos anteriormente, a etapa de carregamento de dados destina-se a guardar todas as contas no formato canónico de cada fonte de dados na tabela Source_Accounts. Mas antes desse carregamento e após a etapa de transformação dos dados, cada rotina trata de guardar os seus resultados num ficheiro CSV contendo informação transformada e classificada sob forma canónica, em vez de inserir diretamente na tabela canónica. Esta operação é ainda dotada de uma ação adicional designada por operação *fix* (conserto). Esta ação é executada apenas quando não é possível processar uma determinada rotina de ETL, por exemplo, devido a um *crash* (falha) durante a comunicação com a base de dados, entre outros. Quando isto acontece, a ação *fix* tenta utilizar um ficheiro CSV produzido por uma sessão anterior (correspondendo à mesma fonte), tentando obter sempre o mais recente ou próximo quando existe.

O nome usado para gerar o ficheiro CSV segue o seguinte formato "csv_[nome da rotina].csv". No fim de todas as rotinas de ETL terminarem e todos os ficheiros CSV produzidos prossegue-se então ao carregamento destes dados para a tabela canónica Source_Accounts.

Na figura 4.10 podemos ver estes ficheiros CSV gerados.

[26] http://rubygems.org/gems/logging

4.6.2 Fase II – Clustering

Na fase de Clustering é realizado o processo de sintetização ou unificação de identidades, descrito no subcapítulo 4.4.

Operação 4: Sintetização de identidades

Para complementar o que foi mencionado anteriormente, é utilizado o pacote lucene-spellchecker-3.4.0.jar[27] para usar a classe JaroWinklerDistance[28], que consiste numa biblioteca de funções de cálculo de distância entre duas *strings*.

4.6.3 Fase III – PosClustering

A fase de PosClustering corresponde ao processamento da solução desenvolvida. Nesta fase são armazenados os resultados processados e devidamente tratados na base de dados de *output*, e ainda um conjunto adicional de informação de estatística acerca de todo o processamento, que será útil para posterior análise. Estes resultados dizem respeito tanto às identidades e contas, como aos valores de NM, NC, BI, PA, AR, AD_USER e AD_EMAIL utilizados pelas identidades.

As operações realizadas nesta fase são:

- Armazenamento de identidades e contas;
- Armazenamento de valores;
- Apresentação de resultados estatísticos.

Operação 5: Armazenamento de identidades e contas

Operação de gravação das identidades sintetizadas para a base da tabela Identity e das contas aplicacionais associadas para a tabela Account. Antes do armazenamento, são calculados alguns atributos que apenas podem ser processados no fim do processo de sintetização, sendo eles os seguintes: AD_Users_Num, AD_Emails_Num, Tainted_Number e Identity_Type.

O atributo Identity_Type tem o valor "Unknown" quando uma identidade está mal constituída, ou seja, caso não possua pelo menos um número de identificação ou quando não possui um nome válido. Esta atribuição permite indicar que uma identidade se encontra incompleta ou fraca, não sendo possível identificar corretamente a personalidade mestre. Ainda, nesta operação são excluídas quaisquer identidades

[27] http://www.java2s.com/Code/Jar/l/Downloadlucenespellchecker340jar.htm

[28] https://lucene.apache.org/core/3_5_0/api/contrib-spellchecker/org/apache/lucene/search/spell/JaroWinklerDistance.html

criadas a partir das fontes de dados do Pulso/Acessos, e todos os valores de AD_Users e AD_Emails são preservados. No entanto, nos restantes atributos multivalor apenas são guardados os dez valores mais recentes.

Operação 6: Armazenamento de valores

Operação de gravação dos valores de Id_NM, Id_NC, Id_BI e AD_Emails provenientes dos respetivos índices invertidos para as respetivas tabelas de saída. Tal como a operação acima, antes do armazenamento procede-se ao cálculo dos seguintes atributos: Recs_Number e Error_Tag (utilizando operações de validação de dados, descritas no subcapítulo 4.5).

É utilizada a gema Ruby Email Veracity[29], que consiste numa biblioteca de funções que permitem verificar se um endereço de *email* é válido ou inválido, e indica o motivo de invalidez.

Operação 7: Apresentação de resultados estatísticos

Operação de produção e apresentação de dados estatísticos acerca do conjunto de identidades sintetizadas e respetivas contas. Esta operação conclui a sessão do processo de sintetização produzindo resultados estatísticos numéricos de todo o procedimento realizado com o objetivo de avaliar e resumir o desempenho da solução desenvolvida. De forma a não comprometer a eficiência do algoritmo decidiu-se apresentar apenas cálculos estatísticos não muito complexos. Assim sendo os cálculos apresentados são os seguintes:

a) Tempo de execução de cada fase (PreClustering, Clustering e PosClustering) e tempo total de execução da solução desenvolvida.

b) Número total de registos iniciais (de entrada), finais (sintetizados) e unificados. Cada registo é entendido como uma identidade ou personalidade mestre usada para efeitos de unificação.

c) Número total de contas aplicacionais processadas.

d) Número total de diferentes valores para cada atributo Id_NM, Id_NC, Id_BI, Id_PA, Id_AR, AD_User e AD_Email utilizados pelas identidades.

Para complementar o estudo estatístico sobre os dados sintetizados foram calculados à posteriori vários relatórios envolvendo diversos gráficos e tabelas relacionando valores numéricos e percentuais. Esta informação encontra-se detalhada no capítulo seguinte de avaliação.

[29] http://rubygems.org/gems/email_veracity

Capítulo 5

Avaliação

Com base no modelo de desenho proposto no capítulo 3 e na sua implementação no capítulo 4 foram realizados diversos testes à solução durante o seu desenvolvimento. Neste capítulo são discutidos detalhadamente o ambiente e os procedimentos adotados em tais testes, apresentando e avaliando os resultados obtidos da plataforma PiasaServer desenvolvida.

5.1 Procedimentos gerais de avaliação utilizados

Os testes de avaliação submetidos ao projeto utilizaram como entrada de dados um grande volume de informação referente a contas do tipo aplicacional apenas e consequentemente as evidências usadas referenciarão somente eventos de *login* ou acessos a aplicações. As aplicações escolhidas para os testes envolvem muitos e poucos elementos de informação que caracterizam as suas contas permitindo assim medir o grau de precisão da solução face aos resultados obtidos.

O projeto foi colocado em avaliação num servidor interno da organização especialmente dedicado onde vários técnicos do departamento onde está inserido este projeto averiguassem a veracidade dos dados resultantes. Durante o desenvolvimento, foram executados testes em diferentes alturas do dia e variando a carga de dados inseridos. Posteriormente convencionou-se que a solução seria executada apenas uma única vez diariamente de madrugada após o procedimento de extração dos *snapshots* das fontes originais e antes do procedimento de captura e associação de eventos às identidades sintetizadas por este projeto.

Ao longo do período de desenvolvimento do projeto foram submetidas diversas melhorias baseadas em comentários e sugestões de implementações contribuindo assim para a avaliação do sistema. Aquando de uma modificação, o projeto foi de imediato disponibilizado para nova execução ou avaliação estando pronto para receber novos comentários e sugestões pelos responsáveis pelo projeto a fim de melhorar a cada iteração. Outro método utilizado para monitorizar o desenvolvimento do projeto

consistiu na realização de reuniões semanais onde foram discutidos diferentes aspetos e formas de implementação com o objetivo de acompanhar e melhorar a solução desenvolvida. Adicionalmente foi também intensamente muito utilizada uma forma que consistiu na seleção de conjuntos de duplicados e após a execução da solução averiguar se tais duplicados foram efetivamente e corretamente unificados ou sintetizados.

5.2 Ambiente de teste

As condições de ambiente de teste usadas para a execução do projeto sem quaisquer problemas necessitaram a utilização de um servidor dedicado com bastante poder de processamento com um total de memória de 64GB e com um sistema operativo Linux de 64 bits. Esta máquina foi utilizada tanto para o desenvolvimento como para os testes realizados, uma vez que é nesta máquina que vai incorporar a solução em ambiente de produção. Após a seleção do *hardware*, procedeu-se à configuração do mesmo onde foram definidas variáveis de ambiente, configuradas e instaladas as ferramentas listadas no subcapítulo 2.4 e outras adicionais necessárias para o funcionamento correto da aplicação, sendo as suas versões as seguintes:

- Java SE Development Kit (JDK) versão 7u17;
- JRuby versão 1.7.3 (com as gemas e bibliotecas externas indicadas no subcapítulo 4.6);
- MySQL edição Community, versão 5.6.10;
- Git versão 1.8.2.

Para medir o grau de eficiência de todos os algoritmos recorreu-se adicionalmente a uma ferramenta *built-in* da linguagem JRuby, designada por Profiler. A utilização desta ferramenta permitiu analisar todas as linhas de código que compõem a aplicação de modo a apresentar no final da sua execução o número de chamadas a determinados métodos, os respetivos tempos de processamento e o tempo total de processamento da aplicação. Desta forma, o Profiler foi extremamente útil para a otimização dos algoritmos implementados sendo utilizado de forma intensiva e melhorando a gestão de memória e consequentemente o tempo total de processamento. A única desvantagem da utilização desta ferramenta é o facto de que o tempo total de processamento da aplicação é aumentado em cerca de 50% a 100%.

5.3 Configurações de teste

O projeto desenvolvido dispõe de um ficheiro de configuração (ficheiro “Global”) onde são definidos vários aspetos importantes para os diferentes componentes que

incorporam o sistema, que facilitam a sua manutenção bem como os testes realizados. Neste ficheiro podemos:

- Definir o caminho a usar para a criação de uma nova sessão de execução do projeto determinando o seu sistema de ficheiros e pastas;
- Definir o formato do cabeçalho a usar durante o processo de ETL para a criação dos ficheiros CSV;
- Definir o caminho de leitura de um ficheiro SQL para criação de novas bases dados PiasaInfo e respetivas tabelas;
- Definir o acesso (credenciais de utilizador) às máquinas e portas de comunicação das bases de dados envolvidas e definir as tabelas utilizadas nos fluxos de dados;
- Definir a ordem das fontes de dados pela qual é realizado o processo de sintetização de identidades.

Além destes aspetos de configuração foram também identificados outros parâmetros que influenciaram o desempenho dos testes realizados. Durante o desenvolvimento do projeto os parâmetros de configuração que mais impacto tiveram foram:

a) O tipo de estrutura de dados usado para representar os *clusters* de identidades num índice invertido. Inicialmente foi testado usando a estrutura de dados Array da linguagem Ruby mas devido à sua lenta gestão quando estão envolvidos grandes volumes de dados modificou-se para uma estrutura de dados do tipo Hash (análoga a uma tabela de dispersão) que permitiu um melhor desempenho face ao uso de Array resultando em rápidas indexações e pesquisas, assim como inserções e remoções. Consequentemente, a solução ficou muito mais eficiente.

b) O método para guardar os dados resultantes do processo ETL. Inicialmente os dados eram inseridos diretamente na base de dados InfoTemp em cada rotina, mas devido ao seu elevado tempo de carregamento (devido à latência da rede usando várias ligações à base de dados) modificou-se e testou-se para outra forma que consistiu em guardar os resultados em ficheiros CSV. Desta forma, esta alteração proporcionou a utilização de apenas uma ligação à base de dados e contribuiu em muito para a redução do tempo de execução da solução, uma vez que os ficheiros são por natureza mais eficientes para escrita e leitura de dados, em vez de bases de dados. Os tempos de processamento passaram em média de 30 segundos para 5 segundos apenas.

c) O método de aplicação do processo de sintetização ou unificação de identidades. Este pode ser aplicado diretamente a todas as 24 rotinas de ETL (resultando em

apenas uma iteração do processo) ou individualmente a cada rotina de ETL (resultando em várias iterações do processo). Nos testes efetuados constatou-se que o método de unificação de uma só vez demonstrou ser menos eficaz do que processado individualmente, pois assim existem mais objetos em memória acabando por sobrecarregar a memória, e consequentemente o aumento do tempo de processamento. A alternativa escolhida passou então por executar a unificação a cada rotina de ETL permitindo assim uma rápido desempenho e possibilitando ainda a criação de uma ordem de unificação, sendo esta muito importante como vimos anteriormente. Esta forma adotada faz uma melhor gestão da memória. De seguida é apresentado um cenário exemplificativo da técnica escolhida envolvendo objetos:

Objetos	Antes	Depois
Identity	240 000	40 000
Account	240 000	240 000
Total	480 000	280 000

Tabela 5.1 - Antes e depois de uma única operação de unificação (método inicial)

Objetos	Antes	Depois (1)	Depois (n)
Identity	60 000	50 000	40 000
Account	50 000	50 000	240 000
Total	110 000	100 000	280 000

Tabela 5.2 – Antes e depois de várias operações de unificação (método escolhido)

Como podemos visualizar, no método escolhido ao aplicar o processo de unificação a cada rotina de ETL apenas os objetos Account vão aumentando. E os objetos Identity vão sendo unificáveis e assim removidos também a cada unificação de rotina de ETL.

d) A adição do índice invertido de nomes ao processo de sintetização de identidades. Uma vez que o algoritmo de unificação desenvolvido é baseado nos princípios do algoritmo Union-Find, este procede sempre à unificação do elemento representativo com cada um dos restantes duplicados de cada *cluster*. Após várias execuções da solução constatou-se que ainda existiam muitos duplicados que poderiam ser combinados mas que não chegavam a ser realizados.

   Por exemplo, segundo o seguinte *cluster* do índice invertido de Id_BI com seis identidades (em que ambas têm o atributo Id_BI em comum): A, B, C, D, E e F. Neste caso, a operação de unificação neste *cluster* vai resultar em 5 tentativas de combinação em que um dos elementos da combinação é sempre o elemento representativo, ou seja, A. No entanto, poderão aqui existir registos que unifiquem

entre si mas não diretamente com o elemento representativo, por exemplo, devido a possuírem nomes totalmente distintos.

Supondo que A unifica com B, C e F mas A não unifica com D e E (devido a estes possuírem diferentes nomes e assim a combinação falha), e ainda D e E unificam entre si com nomes similares. Segundo o desenho inicial da solução, a unificação de D e E não era realizada, resultando em registos duplicados no final.

Este problema foi resolvido utilizando uma de duas abordagens diferentes testadas. Uma solução consistiu em realizar um novo tipo de unificação segundo o atributo FullName, e a alternativa foi proceder a várias combinações dos duplicados de cada *cluster*. Utilizando a unificação por nome, os registos D e E são combinados não segundo o atributo Id_NM mas sim por FullName (caso não exista contradições de valores).

A outra forma consistiu em combinar todos os elementos do *cluster* entre si realizando mais comparações. Segundo o exemplo anterior, as tentativas de unificação realizadas seriam A com B, A com C, …, B com A, e assim até esgotar todos os elementos. Desta forma, o método passa por combinar todos os elementos do *cluster* em que o elemento representativo vai sendo um elemento diferente. Com isto, teríamos no pior caso 120 (5!) tentativas de unificação quando apenas 5 eram realizadas (sem esta implementação). Esta alternativa não chegou a ser viável devido ao aumento do tempo total de execução do processo de cerca de três vezes mais, e assim a solução escolhida foi a unificação por nome aglomerando no mesmo *cluster* registos com nome idênticos.

e) O conjunto de dados de entrada do processo de sintetização de identidades. Inicialmente foi utilizada uma pequena lista de identidades já sintetizadas manualmente mas ainda com muitos duplicados, proveniente de um ficheiro Excel. Posteriormente adaptou-se a solução para utilizar as bases de dados com informação real de origem. Estes testes aplicados a duas diferentes formas de *input* permitiram avaliar a precisão do algoritmo desenvolvido de duas formas diferentes.

f) O desenvolvimento do algoritmo de unificação no processo de sintetização de identidades. Para medir a eficiência deste algoritmo ao longo do desenvolvimento recorreu-se à análise da sua complexidade temporal medindo o tempo que demorou a executar, em que $O(n)$ significa o tempo de execução em função do tamanho *n* da entrada. Inicialmente foram adotadas estratégias quadráticas em que foram processados todos os possíveis pares de dados de entrada resultando num tempo de execução quadrático, $O(n^2)$. Com isto, facilmente se observou que essa não era uma solução viável em tempo útil, e desta forma, foram implementadas

novas melhorias, como por exemplo, os princípios do algoritmo Union/Find reduzindo o problema em subproblemas, resolvendo estes em separado e combinando as suas soluções, resultando assim num tempo de execução, $O(n.\log n)$.

De seguida, podemos observar a evolução do tempo de execução do algoritmo de unificação e o volume de dados envolvidos ao longo do desenvolvimento do projeto:

	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5
Complexidade temporal	$O(n^2)$		$O(n.\log n)$		
Tempo total de unificação	≅ 6 H	≅ 3 H	≅ 30 Min	≅ 12 Min	≅ 7 Min
Quantidade de dados processados	≅ 50 000			≅ 250 000	

Tabela 5.3 - Medição e evolução da eficiência do algoritmo de unificação

5.4 Apresentação de resultados

Os testes realizados permitiram medir o nível de desempenho (eficiência) e de precisão (eficácia) ou qualidade dos resultados da solução em funcionamento em ambiente empresarial. Partindo desses resultados foi possível deduzir um conjunto de dados estatísticos relativos ao desempenho e aos dados de entrada e saída. Os dados a apresentar foram calculados recorrendo ao ficheiro de *log* criado por uma sessão (que se encontra na sua integra no Apêndice A) e para cálculos mais complexos recorrendo a várias consultas à base de dados PiasaInfo. Como descrito no subcapítulo 4.6, este ficheiro de *log* descreve as mensagens de eventos que surgem durante aa execução do projeto incluindo tempos de processamento e quantidades de dados processados.

Os resultados apresentados de seguida recorreram ao uso da ferramenta Profiler e referem a uma execução da plataforma desenvolvida (PiasaServer) num determinado dia e utilizaram como entrada de dados a informação presente na base de dados PiasaSources, sendo esta atualizada diariamente com nova informação. A base de dados de saída PiasaInfo foi apagada totalmente antes no início deste teste.

Resultados obtidos

Os dados apresentados de seguida foram escolhidos como sendo os fatores mais importantes e interessantes que demonstram a qualidade dos dados sintetizados e sua evolução, mas muitos outros poderiam ser apresentados. O método escolhido para apresentar os resultados consistiu em construir tabelas de valores e gráficos que representam essas estatísticas para uma melhor compreensão.

1) Tempos de execução do projeto

Na tabela seguinte são apresentados os tempos de execução da plataforma PiasaServer discriminando os tempos de cada fase e o tempo total de processamento, em segundos e em minutos (em parêntesis).

Fase	Tempo
PreClustering	56 (0.9)
Clustering	421 (7)
PosClustering	226 (3.7)
Total	703 (11.6)

Tabela 5.4 – Tempos de execução

2) Registos processados

Na tabela seguinte estão representadas as quantidades de dados processados que alimentam todo o projeto, nomeadamente os dados de entrada e saída.

Dados	Registos
Identidades iniciais	271 973
Identidades finais (sintetizadas)	80 963
Identidades unificadas	191 010
Contas aplicacionais	118 323

Tabela 5.5 – Registos processados

3) Identidades sintetizadas

Na tabela seguinte estão classificadas as identidades sintetizadas pelo projeto por sendo classificadas por diferentes categorias.

Identidades	Quantidade
Sintetizadas	80 963
Tainted	13 220
Não *tainted*	67 743
Bem formadas	50 044
Mal formadas	30 919
Internas	10 312
Externas	70 651
Presentes na plataforma OIM	55 406
Novas identidades	25 557

Tabela 5.6 - Identidades processadas

As identidades consideradas “mal formadas” são identidades fracas, isto é, identidades que não possuem pelo menos um dos cinco números de identificação obrigatórios ou então não possuem um nome válido. Por outro lado, as identidades consideradas “internas” são aquelas que possuem um número de identificação referindo um número mecanográfico. Ainda, as identidades “presentes na plataforma OIM” dizem respeito a identidades sintetizadas que se encontram nessa plataforma e as “novas identidades” referem novas identidades sintetizadas que não existem atualmente nessa plataforma. Para uma melhor visão e interpretação dos dados acima foram utilizados os seguintes gráficos.

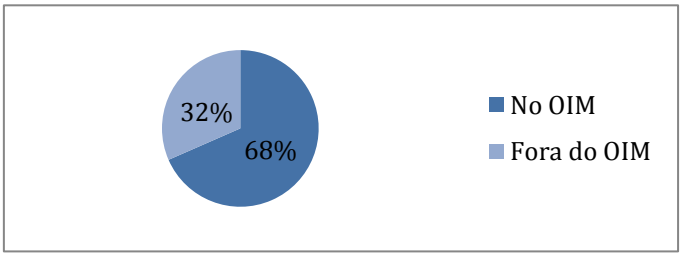

Figura 5.1 – Identidades sintetizadas (1)

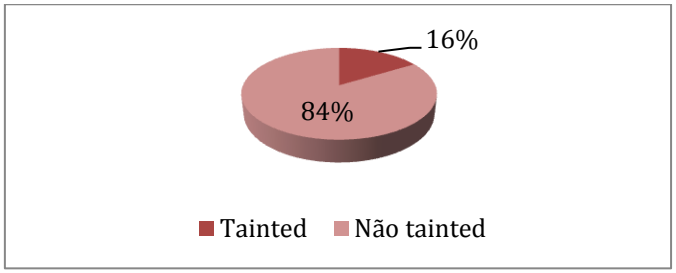

Figura 5.2 - Identidades sintetizadas (2)

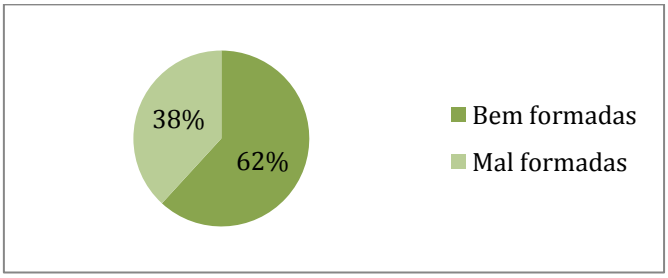

Figura 5.3 - Identidades sintetizadas (3)

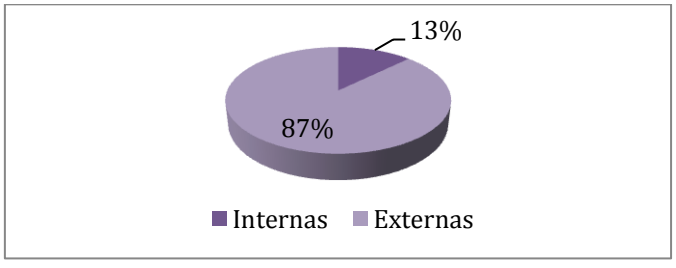

Figura 5.4 - Identidades sintetizadas (4)

4) Relação de identidades por quantidade de unificações

Na tabela seguinte podemos observar a relação entre o número de identidades sintetizadas pelo número de combinações. Foi utilizada uma representação por intervalos para agregar todos os possíveis valores.

Nº. Unificações	Nº. Identidades
0	37788
1 – 5	21863
6 – 9	7661
10 – 49	13012
50 – 99	578
100 – 400	61

Tabela 5.7 – Número de identidades por número de unificações

De modo a facilitar a interpretação da variação destes valores foi construído o seguinte histograma.

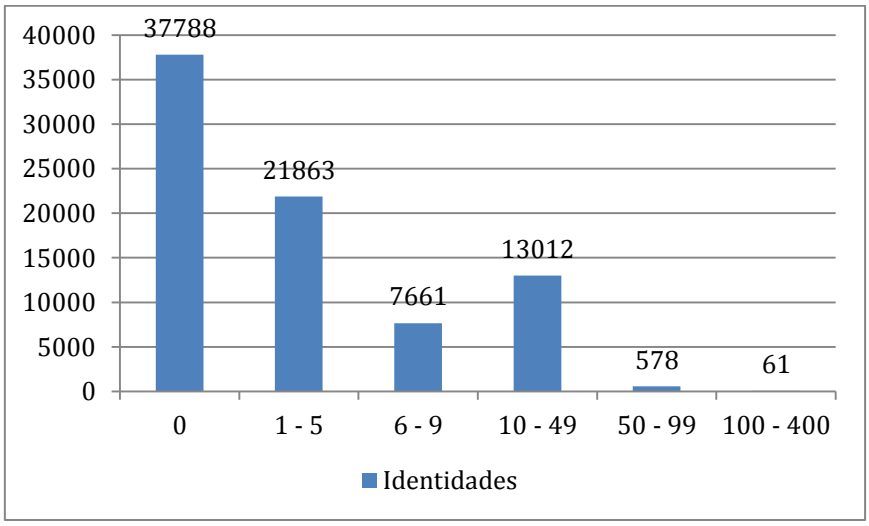

Figura 5.5 – Número de identidades por número de unificações

5) Relação de identidades por quantidade de utilizadores de rede e de endereços de *email*

Na tabela seguinte podemos observar a relação entre o número de identidades sintetizadas pelo número de utilizadores de rede e pelo número de endereços de *email*, uma vez que, é importante perceber a distribuição destes valores. Tal como acima, foi adotada uma representação por intervalos.

De modo a facilitar a interpretação da variação destes valores também foram construídos os seguintes histogramas.

Nº. Utilizadores de rede	Nº. Identidades
0	12 493
1	59 502
2	3 844
3 – 9	4 944
11 – 19	141
20 – 100	39

Tabela 5.8 – Número de identidades por número de utilizadores de rede

Nº de endereços de *email*	Nº. Identidades
0	50 901
1	27 231
2	2 048
3 – 4	692
5 – 9	81
10 – 16	10

Tabela 5.9 – Número de identidades por número de endereços de *email*

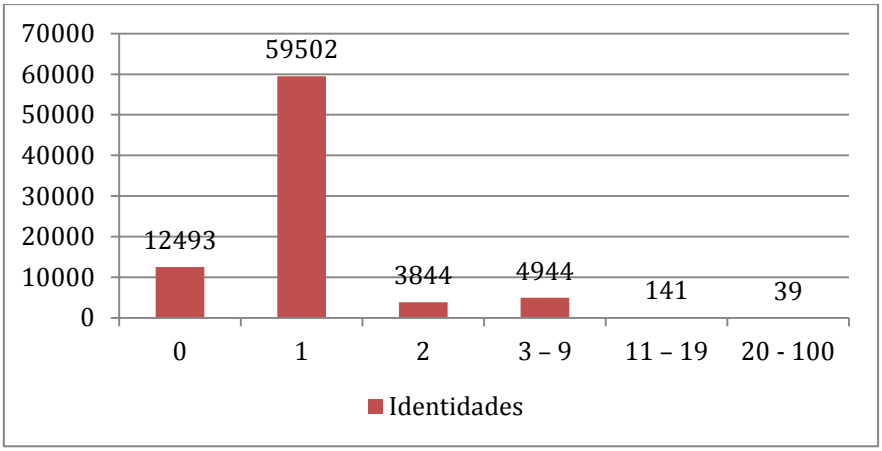

Figura 5.6 – Número de identidades por número de utilizadores de rede

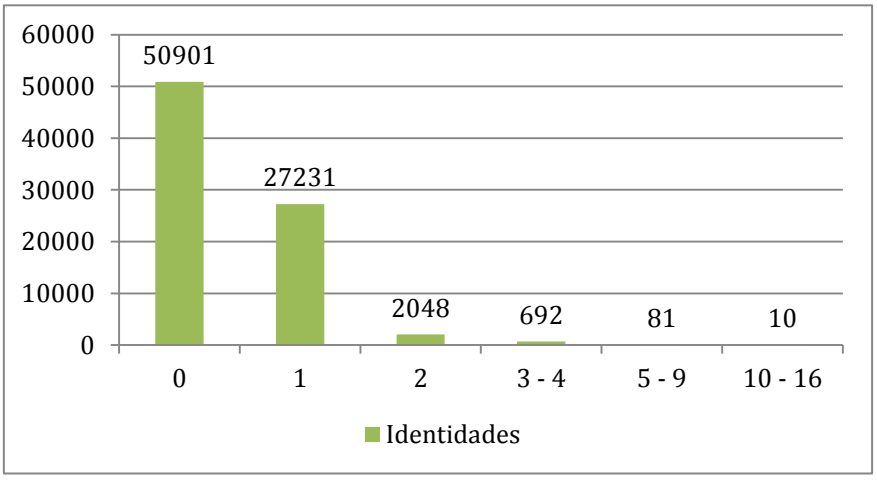

Figura 5.7 - Número de identidades por número de endereços de email

6) Valores processados

Na tabela seguinte estão representadas as quantidades de valores por tipo processados pelo projeto utilizados pelas identidades sintetizadas.

Atributo	Diferentes	Com erros de validação	*Tainted*
Id_NM	12 397	339	491
Id_NC	27 937	549	800
Id_BI	33 702	1 969	738
Id_PA	2 403	-	75
Id_AR	380	-	10
AD_User	76 065	-	5 739
AD_Email	31 772	1 705	1 468
Total	184 656	4 562	9 321

Tabela 5.10 – Valores processados

De modo a facilitar a interpretação da variação destes valores foi também construído o seguinte histograma.

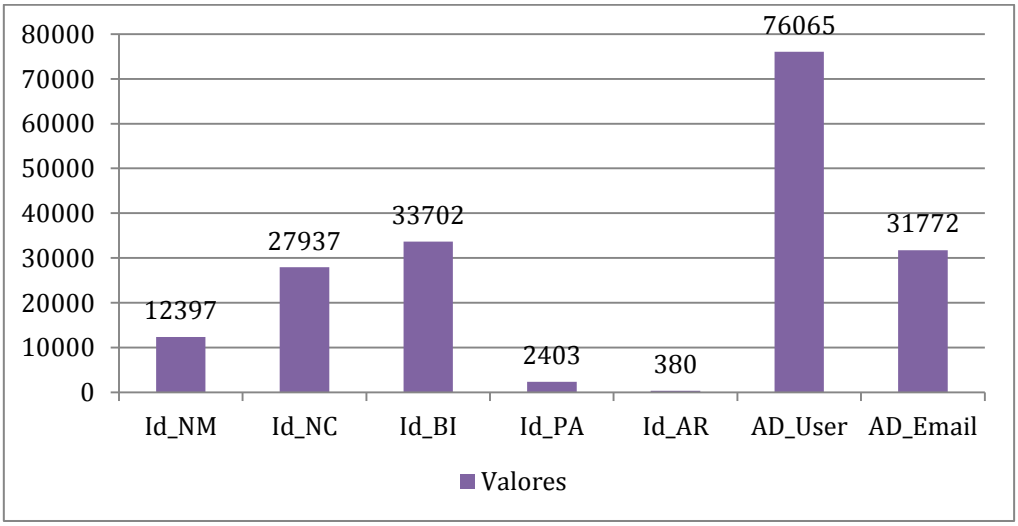

Figura 5.8 - Número de valores por tipo de atributo

5.5 Discussão sobre os testes realizados

Com base nos testes realizados, podemos observar que a fase da plataforma desenvolvida PiasaServer que mais tempo de execução consome é a de sintetização de identidades, pois é neste processo que são realizadas comparações e algoritmos mais complexos. No geral, a solução obteve um ótimo desempenho (cerca de 11.6 minutos) considerando a grande carga de dados processados e os cálculos necessários a realizar face aos tempos inicialmente obtidos. É importante salientar que se constatou que a

comunicação com as bases de dados, nomeadamente as operações de leitura e escrita de dados, contribui para o aumento do tempo total de execução obtido.

Além da medição do desempenho do projeto, também foi muito importante medir a precisão dos resultados obtidos a fim de analisar a veracidade dos mesmos. Considerando a grande quantidade de registos como entrada de dados da solução, o número de identidades sintetizadas encontra-se dentro dos parâmetros esperados, pois a maioria dos registos iniciais unificaram com as fontes de dados provenientes da plataforma OIM e das aplicações. Observou-se ainda que foram criadas novas identidades nos casos em que determinadas aberturas de contas não passam atualmente pela plataforma Pulso/Acessos (e OIM), ou seja, contas que foram criadas antes de aparecer esta plataforma e nunca pediram acesso a outras contas aplicacionais através desta, ou poderão tratar-se de contas criadas diretamente através de consolas, ou seja, contas que continuam a ser criadas fora desta plataforma (quando deviam passar por esta).

No entanto, observou-se por análise que existem 13 mil identidades marcadas como *tainted*, ou seja, identidades que poderiam ser combinadas mas derivado a violações das regras descritas no subcapítulo 4.4.2 não conseguiram unificar, necessitando assim de intervenção humana para resolver esses casos. Isto deve-se ao facto de existir ruído ou "lixo" nos valores obtidos das diversas fontes de dados.

Adicionalmente existe um conjunto considerado de identidades que não possuem pelo menos um dos cinco números de identificação sendo estes imprescindíveis para uma correta identificação da sua personalidade mestre. Ao mesmo tempo constatou-se também que existe um conjunto pequeno de identidades duplicadas que podem ser combinadas não pelo sistema atual mas sim recorrendo ao conceito de similaridade, como será proposto no subcapítulo 6.1.3.

Em relação à distribuição da quantidade de utilizadores de rede e de endereços de *email* podemos observar que existem algumas identidades com um valor que excede o valor permitido pelas normas da organização, ou seja, de um único valor apenas. Nestes casos, cada uma dessas identidades encontra-se devidamente classificada recorrendo aos códigos descritos no subcapítulo 4.4.2.

Por outro lado, em relação aos valores usados que caracterizam uma identidade podemos concluir que o número de Bilhete de Identidade é o mais predominante na organização. No entanto, observamos que existem valores de números de Identificação Fiscal e de Bilhete de Identidade com erros de validação, sendo importante proceder à sua correção devido à elevada importância dos mesmos. Existem também endereços de *email* incorretamente construídos ou com domínios não permitidos pela organização.

Para concluir, a realização deste projeto pretende assim melhorar o desempenho de um analista de segurança através de um conjunto de operações sobre os dados a produzir, proporcionando uma melhor qualidade da mesma e permitindo a identificação das identidades mais prováveis dos utilizadores responsáveis pelas evidências capturadas de acessos a informação reservada. De modo a facilitar a sua função, este tem ao seu dispor várias *flags* (sinalizadores) que classificam as várias identidades e contas, assim como, o campo de anotações com informação relacionada com as falhas de unificações descrevendo detalhadamente o erro encontrado. De futuro, com a integração da componente de captura de eventos e associação destes às identidades sintetizadas será possível ao analista não só analisar os detalhes de identidades e contas, como também identificar situações ilícitas, como por exemplo, acessos a aplicações envolvendo diferentes personalidades mestre.

Ainda, o objetivo deste capítulo era apresentar uma média de várias execuções em vários dias mas não foi possível obter, indicando apenas os resultados de uma única execução da solução. A razão pelo qual não foram disponibilizados mais resultados (provenientes de outras execuções) deve-se ao facto de que a informação processada a cada execução diária não varia muito. Isto, porque para percebermos a evolução dos valores aqui mencionados, a equipa da PTSI teria de proceder diariamente à correção dos casos anómalos identificados pela solução, como por exemplo, a correção de valores *tainted*, e assim, seria possível analisar a evolução destes valores ao longo do tempo mas atualmente esta tarefa ainda não foi atribuída. Desta forma, haveria uma diminuição dos valores *tainted* aproximando-se do valor zero e consequentemente um aumento do número de unificações.

Capítulo 6

Discussão e trabalho futuro

Este capítulo finaliza todo o trabalho realizado para alcançar os objetivos descritos no capítulo 1, e discute aspetos importantes referidos nos capítulos anteriores. São analisadas algumas alternativas de implementação encontradas durante o desenvolvimento e algumas funcionalidades ou melhorias que poderão incorporar a solução num trabalho futuro. Adicionalmente são apresentadas as dificuldades enfrentadas de forma resumida, assim como as opções tomadas, e por fim as considerações finais do projeto face ao plano de trabalhos inicialmente proposto.

6.1 Trabalho futuro

Ao longo do desenvolvimento do projeto foram discutidas várias ideias a implementar no projeto pelos responsáveis do projeto sempre com o objetivo de resolver os problemas indicados no subcapítulo 1.2. As ideias ponderadas foram bastantes, que se traduziram em funcionalidades, onde somente algumas foram efetivamente implementadas, nomeadamente aquelas que demonstraram ser mais importantes, tendo em consideração o tempo e os dados disponíveis a que a solução teve acesso e os requisitos funcionais impostos. De seguida, são apresentadas algumas das funcionalidades que foram propostas e outras sugeridas por mim para serem implementadas num trabalho futuro de modo complementar a solução desenvolvida. Mas inúmeras funcionalidades poderiam ser adicionadas para desenvolvimento consoante aquilo que se pretender.

6.1.1 Aperfeiçoamento do modelo de dados

As melhorias que poderão ser realizadas ao nível do modelo de dados podem ser muitas consoante as necessidades. Contudo, algumas melhorias que considero relevantes e que viriam a fortalecer o modelo atual na caracterização de uma identidade são:

- Inclusão de nova informação que caracteriza a identidade, como por exemplo, informação sobre os dispositivos utilizados aquando de determinado acesso assim como qual a conta acedida nos acessos efetuados. Adicionalmente também será útil informação acerca das localizações lógicas e físicas dos acessos realizados.
- Envolver a gestão de perfis de contas com o objetivo de identificar o nível de risco associado e fazer um processamento destas contas com base nesse aspeto.

6.1.2 Expansão dos dados de entrada

A aplicação foi desenvolvida de modo a tornar-se flexível e facilmente incrementável não só de novas funcionalidades como também de novos dados de entrada, estando o projeto preparado para ser aplicado a qualquer quantidade de registos. Atualmente, o tipo de contas utilizado como entrada de dados do projeto diz respeito exclusivamente a contas aplicacionais (a uma pequeno conjunto de aplicações), e desta forma, as evidências capturadas que fazem uso deste projeto referem portanto acessos a essas aplicações (contas aplicacionais apenas). A melhoria aqui a implementar seria abranger mais aplicações e também outros tipos de acessos ou contas, como contas de sistemas e de base de dados. A convenção a utilizar para adicionar mais fontes de dados é muito simples e requere os seguintes passos: criar pasta na raiz da plataforma com nome da aplicação, criar as respetivas rotinas de ETL, e colocar estas na ordem de unificação do processo de sintetização. Esta melhoria permitiria assim a utilização de outros tipos de evidências, como por exemplo, eventos de torniquetes, entrada e saída de salas, entrada e saída do relógio de ponto, entre outros.

Ainda seria interessante não só trabalhar com contas ativas mas também com contas obsoletas a fim de sintetizar as personalidades mestre desse tipo de contas mesmo que não existam evidências capturadas, a fim de manter uma forma de histórico.

6.1.3 Melhoramento do processo de sintetização

A principal melhoria a implementar no processo de sintetização de identidades seria adicionar o conceito de unificação de valores por similaridade na combinação de registos recorrendo à métrica de distância Jaro-Winkler. Desta forma, o número de unificações poderia aumentar procedendo à unificação de registos que não são unificados com o atual sistema. Assim, o método de deteção de duplicados não seria usado apenas para igualdade exata de valores mas também por similaridade de valores, como por exemplo, dois registos em que um qualquer valor de Id_BI apenas diferisse num único dígito seriam considerados duplicados e portanto viriam a pertencer ao mesmo *cluster* para posterior combinação. O mesmo aconteceria para os restantes índices invertidos.

Outra melhoria a este nível seria estender a validação de dados a outros tipos de atributos de modo a aumentar ainda mais a qualidade de informação envolvida nesta organização. Ainda seria importante alterar a forma de atribuição de um identificador a uma identidade de forma a ser única, isto é, que não altere o valor a cada execução da solução.

6.1.4 Desenvolvimento de um *front-end*

Para complementar o *back-end* desenvolvido seria importante desenvolver uma aplicação *web* que representasse visualmente os dados sintetizados da *datawarehouse* de modo a facilitar a análise desta informação por parte dos analistas de segurança, como por exemplo, a informação apresentada no subcapítulo 5.4. A representação sugerida e ideal seria apresentar as identidades e respetivas contas de acesso sob a forma de grafos de rede, em que teríamos como nós as identidades e contas, e um relacionamento entre ambos estes tipos significaria uma relação.

Esta aplicação permitiria assim visualizar e interagir com as várias personalidades operacionais em redor das suas personalidades mestre (identidades sintetizadas) como com os eventos associados às respetivas contas. Adicionalmente poder-se-ia utilizar o conceito de *timeline* (linha no tempo) para representar os acessos realizados ao longo do tempo.

6.2 Dificuldades enfrentadas

A principal dificuldade enfrentada foi a implementação de operações no desenho da solução de forma eficiente no processamento de grandes quantidades de dados. Por norma, quando se manipula muitos dados existem sempre problemas de desempenho que deverão ser minimizados. Desta forma, existiu sempre o desafio para a máxima otimização possível reduzindo assim a complexidade computacional do algoritmo desenvolvido para este ser executado num tempo viável. Assim, cada nova funcionalidade implementada foi sempre tida em conta o fator eficiência, durante todo o desenvolvimento. Esta dificuldade consumiu constantemente uma grande porção no tempo despendido ao realizar inúmeras otimizações. Outros problemas enfrentados que também marcaram impacto no desenvolvimento do projeto foram os parâmetros de teste descritos no subcapítulo 5.3 na avaliação da solução. Estes problemas passaram pela escolha de uma alternativa quando existiam várias.

Outros problemas mas de menor impacto foi a compreensão inicial do enquadramento da plataforma OIM e da plataforma Pulso, e respetivas relações usadas nas infraestruturas da Portugal Telecom. Também o conhecimento dos procedimentos adotados na criação de contas de acessos, e ainda a familiarização com a linguagem de

programação Ruby que devido ao insuficiente conhecimento desta tecnologia requereu um período de tempo de aprendizagem.

Apesar das dificuldades mencionadas, todas elas foram resolvidas de alguma forma não prejudicando assim o resultado positivo da solução desenvolvida.

6.3 Considerações finais

A solução desenvolvida foi um projeto deveras ambicioso, complexo e abrangente, que sofreu constantemente melhorias e funcionalidades interessantes de modo a enriquecer a estrutura do projeto. A contribuição deste projeto de Gestão de Acessos e Identidades foi positiva para esta organização enriquecendo a atual plataforma OIM com nova informação acerca das identidades já existentes e ainda sobre um conjunto de identidades ausentes nessa plataforma, possibilitando a correta identificação dos proprietários das contas aplicacionais envolvidas, e ainda a identificação de erros de validação de determinados valores. É importante salientar que caso existisse apenas uma fonte de dados com toda a informação necessária não haveria o problema exposto no subcapítulo 1.2, mas em vez disso, existem diferentes fontes caracterizadas por diferentes elementos de informação.

É importante referir que os objetivos propostos na secção 1.2 foram atingidos, e durante o desenvolvimento do projeto existiram alguns desvios ao planeado mas todos eles de forma positiva, procurando-se sempre usar tecnologias e procedimentos que não estavam previstos no arranque do projeto. Alguns desvios traduziram-se em muito tempo de dedicação mas sempre de acordo com os objetivos principais do projeto.

Em relação à linguagem de programa usada, nomeadamente o Ruby foi uma linguagem que me impressionou bastante não só pelas vantagens de desempenho e simplicidade de utilização face a outras linguagens mas também pela sua beleza de estruturação de código. Será com certeza uma linguagem que pretendo usar em futuros projetos por mim desenvolvidos.

Ainda, é importante referir que os nomes e números de documentos de identificação referidos ao longo deste documento são meramente fictícios, não existindo qualquer relação com identidades e contas de acessos verdadeiras.

Apêndice A

Resultados dos testes apresentados

A.1 *Log* da plataforma PiasaServer

INFO Clustering of Identities : ... New clustering session started in 2014-01-07 ...
INFO Clustering of Identities : # Task 1 - Preparation of relational storage system...
INFO Clustering of Identities : ... SQL Database piasa_info_2014_01_07 and tables created ...
INFO Clustering of Identities : # Task 2 - Preparation of the file system...
INFO Clustering of Identities : Created a new directory named 'TEMP/2014-01-07'!
INFO Clustering of Identities : # Task 3 - Processing all ETL routines ...
INFO Clustering of Identities : ... Creating CSV files ...
INFO Clustering of Identities : Done 'etl_accounts_AppE' using source_table 'AppE_users'. Total inserts = 9450. Execution time = 2.991 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppE_extern' using source_table 'AppE_external_users'. Total inserts = 20520. Execution time = 2.403 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppE_pacessos' using source_table 'pacessos_AppE'. Total inserts = 3589. Execution time = 0.467 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppA_pacessos' using source_table 'pacessos_AppA'. Total inserts = 12782. Execution time = 1.614 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppA' using source_table 'oim_AppA_users'. Total inserts = 13259. Execution time = 2.525 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppA_oim' using source_table 'AppA_users'. Total inserts = 13313. Execution time = 1.415 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppC' using source_table 'AppC_users'. Total inserts = 93. Execution time = 0.011 seconds.
INFO Clustering of Identities : Done 'etl_accounts_ad' using source_table 'ad_users'. Total inserts = 45048. Execution time = 5.599 seconds.
INFO Clustering of Identities : Done 'etl_accounts_ad_pacessos' using source_table 'pacessos_ad'. Total inserts = 4733. Execution time = 0.716 seconds.
INFO Clustering of Identities : Done 'etl_accounts_ad_oim' using source_table 'oim_ad_users'. Total inserts = 39897. Execution time = 7.902 seconds.
INFO Clustering of Identities : Done 'etl_accounts_oim' using source_table 'oim_users'. Total inserts = 52849. Execution time = 14.969 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppD' using source_table 'AppD_users'. Total inserts = 62. Execution time = 0.01 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppD_pacessos' using source_table 'pacessos_AppD'. Total inserts = 24. Execution time = 0.004 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppB' using source_table 'AppB_users'. Total inserts = 3872. Execution time = 0.548 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppB_pacessos' using source_table 'pacessos_AppB'. Total inserts = 5994. Execution time = 0.788 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppF' using source_table 'AppF_users'. Total inserts = 6480. Execution time = 0.792 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppG' using source_table 'AppG_users'. Total inserts = 346. Execution time = 0.035 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppG2' using source_table 'AppG2_users'. Total inserts = 209. Execution time = 0.016 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppG_pacessos' using source_table 'pacessos_AppG'. Total inserts = 427. Execution time = 0.038 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppI_oim' using source_table 'oim_AppI_users'. Total inserts = 15015. Execution time = 1.408 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppI_pacessos' using source_table 'pacessos_AppI'. Total inserts = 3204. Execution time = 0.389 seconds.

INFO Clustering of Identities : Done 'etl_accounts_AppI' using source_table 'App_users'. Total inserts = 16207. Execution time = 1.826 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppH_pacessos' using source_table 'pacessos_AppH'. Total inserts = 1877. Execution time = 0.246 seconds.
INFO Clustering of Identities : Done 'etl_accounts_AppH' using source_table 'AppH_users'. Total inserts = 2723. Execution time = 0.267 seconds.
INFO Clustering of Identities : Done all 24 ETL routines!
INFO Clustering of Identities : Total execution time of stage 1: 56.434 seconds (0.9405666666666667) minutes!
INFO Clustering of Identities : # Task 4 - Synthesizing/Unification identities...
INFO Clustering of Identities : ... Running of etls and their clustering individually ...
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_oim.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 37.358 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 10853
INFO Clustering of Identities : Diferentes valores de NCs: 23440
INFO Clustering of Identities : Diferentes valores de BIs: 11455
INFO Clustering of Identities : Diferentes valores de PAs: 1696
INFO Clustering of Identities : Diferentes valores de ARs: 146
INFO Clustering of Identities : Diferentes valores de AD_Users: 0
INFO Clustering of Identities : Diferentes valores de AD_Emails: 19337
INFO Clustering of Identities : Diferentes valores de FullNames: 49844
INFO Clustering of Identities : Identities number (before clustering): 52849
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_oim.csv ...
INFO Clustering of Identities : Unification time: 1.441 seconds
INFO Clustering of Identities : Number of unifications = 2896
INFO Clustering of Identities : Identities number (after clustering): 50111
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_ad_oim.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 28.47 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 10854
INFO Clustering of Identities : Diferentes valores de NCs: 23440
INFO Clustering of Identities : Diferentes valores de BIs: 11469
INFO Clustering of Identities : Diferentes valores de PAs: 1697
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 39764
INFO Clustering of Identities : Diferentes valores de AD_Emails: 22216
INFO Clustering of Identities : Diferentes valores de FullNames: 55803
INFO Clustering of Identities : Identities number (before clustering): 90008
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_ad_oim.csv ...
INFO Clustering of Identities : Unification time: 10.686 seconds
INFO Clustering of Identities : Number of unifications = 22170
INFO Clustering of Identities : Identities number (after clustering): 70820
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppI_oim.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 9.235 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 10854
INFO Clustering of Identities : Diferentes valores de NCs: 23440
INFO Clustering of Identities : Diferentes valores de BIs: 11469
INFO Clustering of Identities : Diferentes valores de PAs: 1697
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 40885
INFO Clustering of Identities : Diferentes valores de AD_Emails: 22216
INFO Clustering of Identities : Diferentes valores de FullNames: 55803
INFO Clustering of Identities : Identities number (before clustering): 85835
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppI_oim.csv ...
INFO Clustering of Identities : Unification time: 4.961 seconds
INFO Clustering of Identities : Number of unifications = 14083
INFO Clustering of Identities : Identities number (after clustering): 71800
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppA_oim.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 9.443 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 10943
INFO Clustering of Identities : Diferentes valores de NCs: 23441
INFO Clustering of Identities : Diferentes valores de BIs: 19400
INFO Clustering of Identities : Diferentes valores de PAs: 1697
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 42369
INFO Clustering of Identities : Diferentes valores de AD_Emails: 22662

INFO Clustering of Identities : Diferentes valores de FullNames: 57880
INFO Clustering of Identities : Identities number (before clustering): 85059
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppA_oim.csv ...
INFO Clustering of Identities : Unification time: 9.255 seconds
INFO Clustering of Identities : Number of unifications = 23198
INFO Clustering of Identities : Identities number (after clustering): 71704
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_ad.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 34.022 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 11863
INFO Clustering of Identities : Diferentes valores de NCs: 23441
INFO Clustering of Identities : Diferentes valores de BIs: 19471
INFO Clustering of Identities : Diferentes valores de PAs: 1764
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 64577
INFO Clustering of Identities : Diferentes valores de AD_Emails: 30424
INFO Clustering of Identities : Diferentes valores de FullNames: 77981
INFO Clustering of Identities : Identities number (before clustering): 116752
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_ad.csv ...
INFO Clustering of Identities : Unification time: 27.931 seconds
INFO Clustering of Identities : Number of unifications = 54790
INFO Clustering of Identities : Identities number (after clustering): 83978
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppE_extern.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 27.678 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 11878
INFO Clustering of Identities : Diferentes valores de NCs: 25002
INFO Clustering of Identities : Diferentes valores de BIs: 29446
INFO Clustering of Identities : Diferentes valores de PAs: 1764
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 70594
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31276
INFO Clustering of Identities : Diferentes valores de FullNames: 86572
INFO Clustering of Identities : Identities number (before clustering): 104498
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppE_extern.csv ...
INFO Clustering of Identities : Unification time: 14.842 seconds
INFO Clustering of Identities : Number of unifications = 30639
INFO Clustering of Identities : Identities number (after clustering): 85784
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppA.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 9.553 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 11878
INFO Clustering of Identities : Diferentes valores de NCs: 25002
INFO Clustering of Identities : Diferentes valores de BIs: 29446
INFO Clustering of Identities : Diferentes valores de PAs: 1764
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 70652
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31276
INFO Clustering of Identities : Diferentes valores de FullNames: 86682
INFO Clustering of Identities : Identities number (before clustering): 99097
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppA.csv ...
INFO Clustering of Identities : Unification time: 14.339 seconds
INFO Clustering of Identities : Number of unifications = 29218
INFO Clustering of Identities : Identities number (after clustering): 84941
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppI.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 11.461 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 11878
INFO Clustering of Identities : Diferentes valores de NCs: 25002
INFO Clustering of Identities : Diferentes valores de BIs: 29446
INFO Clustering of Identities : Diferentes valores de PAs: 1764
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 72950
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31276
INFO Clustering of Identities : Diferentes valores de FullNames: 89448
INFO Clustering of Identities : Identities number (before clustering): 101148
INFO Clustering of Identities : ... Initialize Clustering process ...

INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppI.csv ...
INFO Clustering of Identities : Unification time: 25.203 seconds
INFO Clustering of Identities : Number of unifications = 34211
INFO Clustering of Identities : Identities number (after clustering): 85842
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppE.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 6.608 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12231
INFO Clustering of Identities : Diferentes valores de NCs: 25002
INFO Clustering of Identities : Diferentes valores de BIs: 29448
INFO Clustering of Identities : Diferentes valores de PAs: 1764
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 72951
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31483
INFO Clustering of Identities : Diferentes valores de FullNames: 90063
INFO Clustering of Identities : Identities number (before clustering): 95292
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppE.csv ...
INFO Clustering of Identities : Unification time: 13.243 seconds
INFO Clustering of Identities : Number of unifications = 24034
INFO Clustering of Identities : Identities number (after clustering): 84687
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppF.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 5.301 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12239
INFO Clustering of Identities : Diferentes valores de NCs: 25008
INFO Clustering of Identities : Diferentes valores de BIs: 29542
INFO Clustering of Identities : Diferentes valores de PAs: 1764
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 72986
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31664
INFO Clustering of Identities : Diferentes valores de FullNames: 90063
INFO Clustering of Identities : Identities number (before clustering): 91167
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppF.csv ...
INFO Clustering of Identities : Unification time: 7.023 seconds
INFO Clustering of Identities : Number of unifications = 7429
INFO Clustering of Identities : Identities number (after clustering): 84741
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppH.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 2.066 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12239
INFO Clustering of Identities : Diferentes valores de NCs: 25008
INFO Clustering of Identities : Diferentes valores de BIs: 29542
INFO Clustering of Identities : Diferentes valores de PAs: 1764
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 73037
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31664
INFO Clustering of Identities : Diferentes valores de FullNames: 90839
INFO Clustering of Identities : Identities number (before clustering): 87464
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppH.csv ...
INFO Clustering of Identities : Unification time: 3.379 seconds
INFO Clustering of Identities : Number of unifications = 4238
INFO Clustering of Identities : Identities number (after clustering): 85140
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppG.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 0.252 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12242
INFO Clustering of Identities : Diferentes valores de NCs: 25010
INFO Clustering of Identities : Diferentes valores de BIs: 29544
INFO Clustering of Identities : Diferentes valores de PAs: 1765
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 73037
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31670
INFO Clustering of Identities : Diferentes valores de FullNames: 90917
INFO Clustering of Identities : Identities number (before clustering): 85486
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppG.csv ...
INFO Clustering of Identities : Unification time: 2.456 seconds
INFO Clustering of Identities : Number of unifications = 1866

INFO Clustering of Identities : Identities number (after clustering): 85040
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppG2.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 0.196 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12242
INFO Clustering of Identities : Diferentes valores de NCs: 25010
INFO Clustering of Identities : Diferentes valores de BIs: 29544
INFO Clustering of Identities : Diferentes valores de PAs: 1765
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 73066
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31670
INFO Clustering of Identities : Diferentes valores de FullNames: 91032
INFO Clustering of Identities : Identities number (before clustering): 85249
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppG2.csv ...
INFO Clustering of Identities : Unification time: 2.027 seconds
INFO Clustering of Identities : Number of unifications = 387
INFO Clustering of Identities : Identities number (after clustering): 85051
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppC.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 0.069 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12242
INFO Clustering of Identities : Diferentes valores de NCs: 25010
INFO Clustering of Identities : Diferentes valores de BIs: 29544
INFO Clustering of Identities : Diferentes valores de PAs: 1765
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 73066
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31670
INFO Clustering of Identities : Diferentes valores de FullNames: 91054
INFO Clustering of Identities : Identities number (before clustering): 85144
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppC.csv ...
INFO Clustering of Identities : Unification time: 1.61 seconds
INFO Clustering of Identities : Number of unifications = 208
INFO Clustering of Identities : Identities number (after clustering): 85047
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppD.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 0.046 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12242
INFO Clustering of Identities : Diferentes valores de NCs: 25010
INFO Clustering of Identities : Diferentes valores de BIs: 29544
INFO Clustering of Identities : Diferentes valores de PAs: 1765
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 73066
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31670
INFO Clustering of Identities : Diferentes valores de FullNames: 91063
INFO Clustering of Identities : Identities number (before clustering): 85109
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppD.csv ...
INFO Clustering of Identities : Unification time: 1.618 seconds
INFO Clustering of Identities : Number of unifications = 171
INFO Clustering of Identities : Identities number (after clustering): 85044
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppB.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 2.682 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12242
INFO Clustering of Identities : Diferentes valores de NCs: 25010
INFO Clustering of Identities : Diferentes valores de BIs: 29544
INFO Clustering of Identities : Diferentes valores de PAs: 1765
INFO Clustering of Identities : Diferentes valores de ARs: 149
INFO Clustering of Identities : Diferentes valores de AD_Users: 73066
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31772
INFO Clustering of Identities : Diferentes valores de FullNames: 91183
INFO Clustering of Identities : Identities number (before clustering): 88916
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppB.csv ...
INFO Clustering of Identities : Unification time: 3.493 seconds
INFO Clustering of Identities : Number of unifications = 4720
INFO Clustering of Identities : Identities number (after clustering): 85147
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppA_pacessos.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...

INFO Clustering of Identities : Loading into memory time: 12.281 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12383
INFO Clustering of Identities : Diferentes valores de NCs: 25140
INFO Clustering of Identities : Diferentes valores de BIs: 31440
INFO Clustering of Identities : Diferentes valores de PAs: 2005
INFO Clustering of Identities : Diferentes valores de ARs: 221
INFO Clustering of Identities : Diferentes valores de AD_Users: 75393
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31772
INFO Clustering of Identities : Diferentes valores de FullNames: 91742
INFO Clustering of Identities : Identities number (before clustering): 97929
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppA_pacessos.csv ...
INFO Clustering of Identities : Unification time: 21.098 seconds
INFO Clustering of Identities : Number of unifications = 26351
INFO Clustering of Identities : Identities number (after clustering): 84117
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppI_pacessos.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 2.964 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12387
INFO Clustering of Identities : Diferentes valores de NCs: 25197
INFO Clustering of Identities : Diferentes valores de BIs: 31570
INFO Clustering of Identities : Diferentes valores de PAs: 2032
INFO Clustering of Identities : Diferentes valores de ARs: 240
INFO Clustering of Identities : Diferentes valores de AD_Users: 75401
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31772
INFO Clustering of Identities : Diferentes valores de FullNames: 91860
INFO Clustering of Identities : Identities number (before clustering): 87321
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppI_pacessos.csv ...
INFO Clustering of Identities : Unification time: 7.351 seconds
INFO Clustering of Identities : Number of unifications = 11983
INFO Clustering of Identities : Identities number (after clustering): 82719
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_ad_pacessos.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 4.074 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12387
INFO Clustering of Identities : Diferentes valores de NCs: 27139
INFO Clustering of Identities : Diferentes valores de BIs: 33177
INFO Clustering of Identities : Diferentes valores de PAs: 2118
INFO Clustering of Identities : Diferentes valores de ARs: 326
INFO Clustering of Identities : Diferentes valores de AD_Users: 75883
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31772
INFO Clustering of Identities : Diferentes valores de FullNames: 92239
INFO Clustering of Identities : Identities number (before clustering): 87452
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_ad_pacessos.csv ...
INFO Clustering of Identities : Unification time: 9.353 seconds
INFO Clustering of Identities : Number of unifications = 10796
INFO Clustering of Identities : Identities number (after clustering): 82593
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppE_pacessos.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 3.063 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12396
INFO Clustering of Identities : Diferentes valores de NCs: 27245
INFO Clustering of Identities : Diferentes valores de BIs: 33190
INFO Clustering of Identities : Diferentes valores de PAs: 2152
INFO Clustering of Identities : Diferentes valores de ARs: 335
INFO Clustering of Identities : Diferentes valores de AD_Users: 75931
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31772
INFO Clustering of Identities : Diferentes valores de FullNames: 92493
INFO Clustering of Identities : Identities number (before clustering): 86182
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppE_pacessos.csv ...
INFO Clustering of Identities : Unification time: 8.547 seconds
INFO Clustering of Identities : Number of unifications = 9376
INFO Clustering of Identities : Identities number (after clustering): 82486
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppH_pacessos.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 1.435 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12397
INFO Clustering of Identities : Diferentes valores de NCs: 27264

INFO Clustering of Identities : Diferentes valores de BIs: 33223
INFO Clustering of Identities : Diferentes valores de PAs: 2389
INFO Clustering of Identities : Diferentes valores de ARs: 351
INFO Clustering of Identities : Diferentes valores de AD_Users: 75932
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31772
INFO Clustering of Identities : Diferentes valores de FullNames: 92756
INFO Clustering of Identities : Identities number (before clustering): 84363
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppH_pacessos.csv ...
INFO Clustering of Identities : Unification time: 3.979 seconds
INFO Clustering of Identities : Number of unifications = 4514
INFO Clustering of Identities : Identities number (after clustering): 82473
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppG_pacessos.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 0.313 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12397
INFO Clustering of Identities : Diferentes valores de NCs: 27271
INFO Clustering of Identities : Diferentes valores de BIs: 33228
INFO Clustering of Identities : Diferentes valores de PAs: 2390
INFO Clustering of Identities : Diferentes valores de ARs: 351
INFO Clustering of Identities : Diferentes valores de AD_Users: 75932
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31772
INFO Clustering of Identities : Diferentes valores de FullNames: 92760
INFO Clustering of Identities : Identities number (before clustering): 82900
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppG_pacessos.csv ...
INFO Clustering of Identities : Unification time: 2.659 seconds
INFO Clustering of Identities : Number of unifications = 1859
INFO Clustering of Identities : Identities number (after clustering): 82452
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppD_pacessos.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 0.03 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12397
INFO Clustering of Identities : Diferentes valores de NCs: 27272
INFO Clustering of Identities : Diferentes valores de BIs: 33228
INFO Clustering of Identities : Diferentes valores de PAs: 2390
INFO Clustering of Identities : Diferentes valores de ARs: 351
INFO Clustering of Identities : Diferentes valores de AD_Users: 75932
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31772
INFO Clustering of Identities : Diferentes valores de FullNames: 92762
INFO Clustering of Identities : Identities number (before clustering): 82476
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppD_pacessos.csv ...
INFO Clustering of Identities : Unification time: 1.634 seconds
INFO Clustering of Identities : Number of unifications = 233
INFO Clustering of Identities : Identities number (after clustering): 82452
INFO Clustering of Identities : ... Run ETL routine: csv_etl_accounts_AppB_pacessos.csv ...
INFO Clustering of Identities : ... Done loading accounts from Source_Accounts table into memory ...
INFO Clustering of Identities : Loading into memory time: 4.967 seconds
INFO Clustering of Identities : Diferentes valores de NMs: 12397
INFO Clustering of Identities : Diferentes valores de NCs: 27937
INFO Clustering of Identities : Diferentes valores de BIs: 33702
INFO Clustering of Identities : Diferentes valores de PAs: 2403
INFO Clustering of Identities : Diferentes valores de ARs: 380
INFO Clustering of Identities : Diferentes valores de AD_Users: 76065
INFO Clustering of Identities : Diferentes valores de AD_Emails: 31772
INFO Clustering of Identities : Diferentes valores de FullNames: 92823
INFO Clustering of Identities : Identities number (before clustering): 88446
INFO Clustering of Identities : ... Initialize Clustering process ...
INFO Clustering of Identities : ... Done clustering applied to ETL: csv_etl_accounts_AppB_pacessos.csv ...
INFO Clustering of Identities : Unification time: 6.69 seconds
INFO Clustering of Identities : Number of unifications = 12150
INFO Clustering of Identities : Identities number (after clustering): 81809
INFO Clustering of Identities : Total execution time of stage 2: 421.303 seconds (7.021716666666666) minutes!
INFO Clustering of Identities : # Task 5 - Storing identities and accounts into output database (Piasa_Info)...
INFO Clustering of Identities : ...saved identity table ...
INFO Clustering of Identities : ...saved account table ...
INFO Clustering of Identities : # Task 6 - Storing values into output database (Piasa_Info)...
INFO Clustering of Identities : ...saved list_nm table ...
INFO Clustering of Identities : ...saved list_nc table ...
INFO Clustering of Identities : ...saved list_bi table ...

```
INFO  Clustering of Identities : ...saved list_pa table ...
INFO  Clustering of Identities : ...saved list_ar table ...
INFO  Clustering of Identities : ...saved list_ad_user table ...
INFO  Clustering of Identities : ...saved list_ad_email table ...
INFO  Clustering of Identities : ### Done Clustering ###
INFO  Clustering of Identities : Total execution time of stage 3:  226.383 seconds (3.77305) minutes!
INFO  Clustering of Identities : # Task 7 - Presentation of statistical results ...
INFO  Clustering of Identities : Execution time of stage 1 (PreClustering): 56.434 seconds
INFO  Clustering of Identities : Execution time of stage 2 (Clustering): 421.303 seconds
INFO  Clustering of Identities : Execution time of stage 3 (PosClustering): 226.383 seconds
INFO  Clustering of Identities : Total of execution time: 704.12 seconds
INFO  Clustering of Identities : Number of initial identities: 271973
INFO  Clustering of Identities : Number of final identities: 81809
INFO  Clustering of Identities : Number of unified identities: 190164
INFO  Clustering of Identities : Number of applicational accounts: 118323
INFO  Clustering of Identities : Number of different values of Id_NMs: 12397
INFO  Clustering of Identities : Number of different values of Id_NCs: 27937
INFO  Clustering of Identities : Number of different values of Id_BIs: 33702
INFO  Clustering of Identities : Number of different values of Id_PAs: 2403
INFO  Clustering of Identities : Number of different values of Id_ARs: 380
INFO  Clustering of Identities : Number of different values of AD_Users: 76065
INFO  Clustering of Identities : Number of different values of AD_Emails: 3177
```

Bibliografia

[1] José A. S. Alegria, Tiago F. R. Carvalho, e Ricardo G. Ramalho. "Uma experiência *open source* para "tomar o pulso" e "ter pulso" sobre a função sistemas e tecnologias de informação", 2005

[2] João Martins. "Investigação e Desenvolvimento de um sistema automático de mapeamento e visualização de ativos numa rede empresarial de grande dimensão", Tese de Mestrado, Faculdade de Ciências da Universidade de Lisboa, 2012.

[3] The Neo4j Team. "The Neo4j Manual 1.7", Neo Technology, 2012.

[4] Pedro Veiga. "Segurança Informática", Faculdade de Ciências da Universidade de Lisboa e FCCN, Outubro de 2006

[5] Lise Getoor e Ashwin Machanavajjhala. "Entity Resolution", University of Maryland e Duke University, 2012.

[6] Luciana Costa. "What is Sarbanes-Oxley and the impacts on IT", 2006

[7] Ana Paula Joaquim Gonçalves. "Gestão de Identidades em ambientes UNIX", Projecto Final de Licenciatura, Universidade Atlântica, 2011

[8] The Open Group. "Identity Management", 2004

[9] Steven Euijong Whang, David Marmaros, Hector Garcia-Molina, "Pay-As-You-Go Entity Resolution," IEEE Transactions on Knowledge and Data Engineering, 2012.

[10] Oracle Corporation. "White Paper Oracle, Introdução ao Oracle Identity Management", 2008

[11] Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis and Vassilios S. Verykios. "Duplicate Record Detection: A Survey", 2007

[12] Nick Koudas, Sunita Sarawagi and Divesh Srivastava. "Record Linkage: Similarity Measures and Algorithms", 2006

[13] Ioana Bazavan Justus. "Identity Management Series", Security Catalyst, 2010

[14] Margaret Rouse, "Identity Management", 2008

[15] John K. Waters, “The ABCs of Identity Management”, 2005

[16] William E. Yancey, “Evaluating String Comparator Performance for Record Linkage”, 2005

[17] Microsoft Corporation. “Business Ready Security Identity and Access Management Solution”, Microsoft Forefront, 2009

[18] Sergio Miranda Freire, Rita de Cássia Braga Gonçalves, André Cipriani Bandarra. “Análise da Efetividade de Comparadores de Strings para Discriminar Pares Verdadeiros de Pares Falsos no Relacionamento de Registros”, 2009

[19] Maria Estela Vieira da Silva, “XSimilarity: Uma ferramenta para consultas por similaridade embutidas na linguagem XQuery”, 2007

[20] William E. Winkler. “Overview of Record Linkage and Current Research Directions”, 2006

[21] WilliamW. Cohen, Pradeep Ravikumar e Stephen E. Fienberg. “A Comparison of String Distance Metrics for Name-Matching Tasks”, 2003

[22] Professor Jorge Picado. “A álgebra dos sistemas de identificação: da aritmética modular aos grupos diedrais”, 2001

[23] Dulce Domingos. “Segurança”, Faculdade de Ciências da Universidade de Lisboa, 2011

[24] Chirayu Poundarik. “ETL Process in Data Warehouse”, Georgia State University, 2013

[25] Alvaro E. Monge e Charles P. Elkan. “An eficiente domain-independent algorithm for detecting approximately duplicate database records”, Department of Computer Science and Engineering, University of California, San Diego, 1997

[26] Avrim Blum. “Graph Algorithms: Union-find.”. Department of Computer Science, Carnegie Mellon University, 2007